

JPRS-UCC-85-006

5 September 1985

# USSR Report

CYBERNETICS, COMPUTERS AND  
AUTOMATION TECHNOLOGY



FOREIGN BROADCAST INFORMATION SERVICE

NOTE

JPRS publications contain information primarily from foreign newspapers, periodicals and books, but also from news agency transmissions and broadcasts. Materials from foreign-language sources are translated; those from English-language sources are transcribed or reprinted, with the original phrasing and other characteristics retained.

Headlines, editorial reports, and material enclosed in brackets [] are supplied by JPRS. Processing indicators such as [Text] or [Excerpt] in the first line of each item, or following the last line of a brief, indicate how the original information was processed. Where no processing indicator is given, the information was summarized or extracted.

Unfamiliar names rendered phonetically or transliterated are enclosed in parentheses. Words or names preceded by a question mark and enclosed in parentheses were not clear in the original but have been supplied as appropriate in context. Other unattributed parenthetical notes within the body of an item originate with the source. Times within items are as given by source.

The contents of this publication in no way represent the policies, views or attitudes of the U.S. Government.

PROCUREMENT OF PUBLICATIONS

JPRS publications may be ordered from the National Technical Information Service (NTIS), Springfield, Virginia 22161. In ordering, it is recommended that the JPRS number, title, date and author, if applicable, of publication be cited.

Current JPRS publications are announced in Government Reports Announcements issued semimonthly by the NTIS, and are listed in the Monthly Catalog of U.S. Government Publications issued by the Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402.

Correspondence pertaining to matters other than procurement may be addressed to Joint Publications Research Service, 1000 North Glebe Road, Arlington, Virginia 22201.

Soviet books and journal articles displaying a copyright notice are reproduced and sold by NTIS with permission of the copyright agency of the Soviet Union. Permission for further reproduction must be obtained from copyright owner.

5 September 1985

## USSR REPORT

## CYBERNETICS, COMPUTERS AND AUTOMATION TECHNOLOGY

## CONTENTS

## GENERAL

Artificial Intelligence: Is It Possible? (Germogen Sergeyevich Pospelov, Interview; ZHURNALIST, No 7, Jul 84).....	1
Leningrad Computer Repair Firms Fail To Give Needed Services (V. Tveritina, V. Chichin; LENINGRADSKAYA PRAVDA, 22 May 85).....	8
Need for Computers, Automation in Production Cited (I. Glebov; SOTSIALISTICHESKAYA INDUSTRIYA, 2 Apr 85)....	10
Need for Microcalculators, New Newspaper Column Given (I. Danilov; SOTSIALISTICHESKAYA INDUSTRIYA, 16 May 85)..	13
Difficulties in Factory Automation Told (P. Derunov; PRAVDA, 7 May 85).....	16
Computer in a Box (A. Ivanov; MOSKOVSKAYA PRAVDA, 9 Feb 85).....	20
Standard Design - the Basis of Creation of the Automated System of Financial Calculations of the Russian Federation (B. I. Filimonov, A. P. Kolesnik, et al.; FINANCY SSSR, No 4, Apr 85).....	20

## HARDWARE

Prospects for Creation of Optical Digital High-Performance Computers (V. M. Yegorov, E. G. Kostsov; AVTOMETRIYA, No 1, Jan-Feb 85).....	22
--	----

'Start' Consortium Pushes Mars, Fifth-Generation Development (SOVETSKAYA ESTONIYA, 14 Jun 85).....	37
'Disk' Complex Helps Interpret Weather Satellite Data (V. Reznik; IZVESTIYA, 10 Jun 85).....	38
Elektronika MS 1603 High-Speed Peripheral Processor (V. A. Dyboy, V. V. Kashtanov, et al.; AVTOMETRIYA, No 2, Mar-Apr 85).....	39
The 15UT-4-017 Complex as a Workstation for Circuit Engineering Modeling (V. Ye. Mezhov; AVTOMETRIYA, No 2, Mar-Apr 85).....	44
Diagnostics of Peripheral Processors and Controllers (V. A. Dyboy, O. S. Semenova, et al.; AVTOMETRIYA, No 2, Mar-Apr 85).....	48
An Approach to Implementing CAMAC Microprocessor Modules (Ye. M. Bazarnyy, A. N. Vystavkin, et al.; UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 2, Mar-Apr 1985). .	54
Compact, High-Speed Algorithm for Laying Out Printed Circuit Board Runs (D. Ye. Zapolotskiy; UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 2, Mar-Apr 1985).....	59
Operating Algorithm for Adaptive Transmission System (A. B. Aleksandrov, Yu. G. Dolotin; AVTOMETRIYA, No 2, Mar-Apr 85).....	68
Interface Between YeS Computers, Sensors and Analog Signal Recorders (S. V. Bondovskiy, V. V. Deyev, et al.; UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 1, Jan 85).....	68
<b>SOFTWARE</b>	
Classification of Real-Time Program Testing Tasks and Methods (I. N. Dolganyuk, Ye. Ya. Karpovskiy; UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 2, Mar-Apr 1985). .	70
DBS/R Database Control System (Yu. Sharr; UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 2, Mar-Apr 1985).....	84
Some Problems in the Technology of Creation and Introduction of Databases on Machine-Readable Media (Zh. F. Sergazin, V. V. Tolochko; KLASSIFIKATORY I DOKUMENTY, No 12, Dec 84).....	93

Formalized Statement of Problems of Systems Design of Terminal System Software (V. V. Pirogov, V. N. Batrak; AVTOMATIKA I VYCHISLITEL'NAYA TEKHNIKA, No 4, Jul-Aug 84).....	93
Modeling of Associative Memory for Relational Databases (O. R. Frolov; PROGRAMMIROVANIYE, No 6, Nov-Dec 84).....	94
Some Problems of Theory of Relational Data Processing Languages (L. V. Vernik, I. M. Vinitskiy; KIBERNETIKA, No 5, Aug-Sep 84).....	94
Message Output Control in the INES System (A. N. Godunov, N. Ye. Yemel'yanov, et al.; PROGRAMMIROVANIYE, No 6, Nov-Dec 84).....	95
Complexity of Structural Testing of Program Modules With Loops (V. V. Lipayev, B. A. Pozin; PROGRAMMIROVANIYE, No 6, Nov-Dec 84).....	95
Principles of Design of a Toolkit for Development of Software for Automation and Control Systems Based on SM Computers (V. V. Kibitkin; UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 1, Jan 85).....	96
Graphics Software System for Machine Graphics System (S. A. Pozdneyev; PROGRAMMIROVANIYE, No 6, Nov-Dec 84)...	97
Implementation of the Grass Graphics Software in the SM Computer Environment (M. A. Kurilov; UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 1, Jan 85).....	97
Presentation of Graphic Information on RIN-609 Display Screen (L. G. Avetisov, I. Ye. Vasinyuk, et al.; UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 1, Jan 85).....	98
Assembler 2 Programming System for YeS OS (V. S. Voyush, G. S. Degtyareva, et al.; PROGRAMMIROVANIYE, No 6, Nov-Dec 84).....	98
Translation of APL System Statements for the SM-1 Computer (A. V. Kondrashev; UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 1, Jan 85).....	99
Development of the FORT-P Problem-Oriented Language (V. A. Smol'nikov; UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 1, Jan 85).....	99

SORT-7/SM Sorting Subsystem (K. P. Kvachuk; UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 1, Jan 85).....	100
Data Models in Dialogue Integrated Automatic Planning System for Industrial Entities (S. B. Dodonov, V. A. Visikirskiy; KIBERNETIKA, No 5, Aug-Sep 84).....	100
Games Modeling of Dialogue in Interactive Systems (S. A. Yushkov; UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 1, Jan 85).....	101
Design of the Base Language for an Information Machine (A. K. Aylamazyan, M. M. Gilula; DOKLADY AKADEMII NAUK SSSR, No 2, Jan 85).....	102
Language Implementation of Parallel Asynchronous Computation Model (T. I. Lel'chuk; KIBERNETIKA, No 5, Aug-Sep 84).....	102
Coordinate Transformation Algorithms for Microprocessors (N. S. Anishin; PRIBOROSTROYENIYE, No 4, May 85).....	103
<b>APPLICATIONS</b>	
Shipboard Complex Processes Information From Weather Satellites (VODNYY TRANSPORT, 25 Jun 85).....	104
Collective Use Computer System for the Academy of Sciences KaSSR (U. M. Sultangazin, I. T. Pak, et al.; UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 2, Mar-Apr 1985).....	105
"New" Motor for Robots Described (I. Demchenko; SOTSIALISTICHESKAYA INDUSTRIYA, 16 May 85).....	112
Algorithms for Reproducing Tool Movement Paths for Program- Controlled Equipment (V. M. Vodovozov; UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 2, Mar-Apr 1985).....	114
Minimizing Image Formation Times on a Drafting/Graphic Automaton (V. V. Kravchuk; AVTOMETRIYA, No 2, Mar-Apr 85).....	120

Digital Processing of Images in Tracking a Moving Object (B. A. Alpatov, A. A. Selyayev, et al.; PRIBOROSTROYENIYE, No 2, Feb 85).....	123
Status and Prospects for Development of Gas Industry Automated Control (GAZOVAYA PROMYSHLENNOST': SERIYA AVTOMATIZATSIIA, TELEMECHANIZATSIIA I SVYAZ' V GAZOVYY PROMYSHLENNOSTI, No 5, Sep 84).....	128
'ASU-Inspeksiya' Automated Information Processing System (A. G. Budanov, T. D. Orlova; FINANSY SSSR, No 10, Oct 84).....	129
Information Support of the Central USSR State Planning Commission Automated Plan Calculation System Central Group of Problems (E. F. Penchik; KLASSIFIKATORY I DOKUMENTY, No 12, Dec 84).....	129
Design of Republic State Planning Commission Automated Plan Calculation System Databases (R. V. Soms, I. A. Blum, et al.; KLASSIFIKATORY I DOKUMENTY, No 10, Oct 84).....	130
Development of the Republic Assortment Portion of the All-Union Classifier for Industrial and Agricultural Products in the Moldavian SSR (Z. G. Kozina, N. A. Kristya, et al.; KLASSIFIKATORY I DOKUMENTY, No 10, Oct 84).....	131
Semantic-Syntactic Natural Language Phrase Analyzer in Learning Dialogue Information Processing Systems (T. P. Dovgyallo; UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 1, Jan 85).....	131
Field Computer System Based on 'Elektronika 100-25' Minicomputer and 'Elektronika MT-70' Peripheral Processor (F. I. Larin, V. Ye. Mezhov, et al.; AVTOMETRIYA, No 6, Nov-Dec 84).....	132
Automated System for Measurement, Analysis and Processing of Experimental Data at the Institute of High Energy Physics, Kazakh Academy of Sciences (M. A. Tashimov, I. Ya. Chasnikov; UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 1, Jan 85).....	133
One-Dimensional Statistical Analysis of Geometric Heights on Glide Path (I. V. Khairakulov, A. V. Sheyman, et al.; PRIKLADNAYA MATEMATIKA I MEKHANIKA, 1983).....	133

Information Support for System for Processing Remote and Contact Measurements of Environmental Parameters (T. M. Askerov, A. A. Agayev; IZVESTIYA AKADEMII NAUK AZERBAYDZHANSKOY SSR: SERIYA FIZIKOTEKHNICHESKIKH I MATEMATICHESKIKH NAUK, No 3, 1984).....	134
The 'SKALD' Program - Experience in Modeling Poetic Creativity for a Computer (A. M. Kondratov, A. V. Zubov; KIBERNETIKA, No 5, Aug-Sep 84).....	134
Analog and Digital Astronomical Image Processing (O. Ya. Usykov, V. M. Dudinov, et al.; VISNYK AKADEMIYI NAUK UKRAYINSKOYI RSR, No 2, Feb 85).....	135

#### NETWORKS

Integrated Network System for Automating Scientific Research (O. S. Zudin: AVTOMATIKA I VYCHISLITEL'NAYA TEKHNIKA, No 4, Jul-Aug 84).....	136
Structure of High-Speed YeS Computer-Based Workstation for a Local Area Network (V. A. Red'ko, V. A. Gobzemis; AVTOMATIKA I VYCHISLITEL'NAYA TEKHNIKA, No 4, Jul-Aug 84).....	136
Communications Subnetworks of Local Area Computer Networks (E. A. Yakubaytis; AVTOMATIKA I VYCHISLITEL'NAYA TEKHNIKA, No 6, Nov-Dec 84).....	137
Principles of Design of Local Area Networks Using Universal Microprocessors (B. N. Malinovskiy, A. I. Nikitin, et al.; UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 1, Jan 85).....	138

#### THEORY OF COMPUTATIONS

Search for Optimal Guaranteed Solutions of Large Problems in Semi-Infinite Programming (O. A. Zaboyeva, S. G. Timokhin, et al.; ZHURNAL VYCHISLITEL'NOY MATEMATIKI I MATEMATICHESKOY FIZIKI, No 1, Jan 85).....	139
--	-----

#### EDUCATION

Computer Use in Grade School Cited (A. Andrusenko; SOVETSKAYA KULTURA, 2 Apr 85).....	140
--	-----

Call for Mass Computer-Literacy Training in Latvia  
(E. Yakubaytis; SOVETSKAYA LATVIYA, 21 May 85)..... 142

ORGANIZATIONS

UNESCO Center for Engineering Education Automation in Tbilisi  
(ZARYA VOSTOKA, 7 Jun 85)..... 143

GENERAL

ARTIFICIAL INTELLIGENCE: IS IT POSSIBLE?

Moscow ZHURNALIST in Russian No 7, Jul 84 pp 74-75, 78

[Interview with Germogen Sergeyevich Pospelov, corresponding member, USSR Academy of Sciences, chairman, USSR Academy of Sciences' Scientific Council on the Problem "Artificial Intelligence," by Yu. Samoylov, candidate of technical sciences, scientific reviewer, MOSKOVSKIYE NOVOSTI; date and place not specified]

[Text] The tempestuous development of computers has enabled them to intrude into areas that have always been regarded as belonging to man alone. There have appeared programs that provide these "iron helpers" with the capabilities to play chess, compose poetry and music, draw, translate from one language to another, control technological processes and do much more than not long ago would have seemed fantastic. The huge successes achieved in cybernetics have engendered in many people--including some well-known specialists--the certitude that an "artificial intelligence" that will surpass man's will be created in the very near future. The press has contributed greatly to the dissemination of such ideas. But are they justified?

What we can expect from "intelligent" machines and what we cannot is the subject of a conversation between Candidate of Technical Sciences Yu. Samoylov, scientific reviewer for the newspaper MOSKOVSKIYE NOVOSTI, and USSR Academy of Sciences Corresponding Member Germogen Pospelov, chairman of the USSR Academy of Sciences' Scientific Council on the Problem "Artificial Intelligence."

[Question] There are quite a few definitions of "artificial intelligence." They frequently contradict each other...

[Answer] And not only definitions, but also unhealthy sensations around this term. Many people perceive it in the literal meaning of the word, as a synonym for an artificial mind, which leads to incorrect opinions that, unfortunately, are widely reflected in the press and can do nothing but harm. In reality, there can be no talk of an artificial mind.

Artificial intelligence is a figurative, metaphorical title for an entire scientific field that brings together mathematicians, programmers, linguists, psychologists, logicians, engineers and many other specialists. The basic goal of this field is to simulate man's creative activity (in the sense of the final results) on computers.

[Question] Germogen Sergeyevich, do you assume that artificial intelligence, as it is understood in the literal sense, cannot be created?

[Answer] It is fundamentally impossible. The human brain is an extraordinarily complex system that consists of about 10-15 billion interconnected cells. Each cell is a complex system in and of itself. There exists a quite plausible hypothesis, according to which an individual cell processes the signals entering it in a manner similar to that of computer. Thus, even a perfect machine, which can only be imagined, cannot compare with the human brain.

There is yet another problem that makes it impossible to give an affirmative answer. Here I am talking about the discovery at the beginning of the 1970's of the different functions of the human brain's two hemispheres. Each hemisphere has its own method of thinking, and it is completely different from the other's. Roughly speaking, one hemisphere thinks logically and the other in patterns. Our thinking is based on two types of perception of the outside world: sensual unconscious and conscious. What a human being perceives and, as it came to be, can express in words, is only a small part of the brain's total functioning. Thought processes cannot be observed directly. We can only form an opinion about them indirectly, by studying how the information entering the brain is transformed. Therefore, very little is known about what goes on in the "pattern" hemisphere. And without this we cannot create intelligence in any way resembling man's. The world of emotions and inclinations and unclear subconscious aims, which play a huge role in human thought and behavior, are inaccessible to us. It may be that in the distant future we will succeed in building machines that are capable of approaching human intelligence to some degree. But this will become possible only if there occurs a revolution in computer technology that is not now foreseeable and people learn how to make computer complex on the elementary level, with billions of these complex elements.

[Question] Let us take ourselves to that distant future for a minute. Some doctor-scientists think that an artificial intelligence endowed with senses, emotions, a conscious and a subconscious, will be regarded as a being having all the rights of a human being. The activity of one or another of those senses can be changed only if it agrees.

[Answer] I would prefer not to discuss projects that are too fantastic and that, from my point of view, cannot be realized in the foreseeable future. Without sufficient grounds, some specialists are endowing machines with human qualities. This is not new, in principle, and has deep roots in the history of culture. We encounter the animation of objects in stories and legends. However, science is not a story. A machine, no matter how perfect it is, is dead matter. It cannot have senses, or emotions, or needs, or desires--

all those things with which man is endowed. Can you imagine a machine experiencing love for someone? I, for one, cannot. And how would it be with that vital experience that man acquires in society by clashing constantly with purely human problems?

[Question] This means that we should not talk about a machine having thoughts at some time or another?

[Answer] In general, machines cannot think, either logically or in patterns. This property belongs only to living, highly organized matter.

[Question] Nevertheless there are people, including well-known scientists, who think completely seriously that an artificial intelligence will not only be created, but that its capacity will surpass man's. And they even give a date: the end of the 20th Century. How can such predictions be explained?

[Answer] As a rule, such statements are made only by people who are unfamiliar with science or by specialists who are not dealing specifically with our problem and have an extremely superficial view of it. Even in our field, however, one encounters indefatigable optimists. I think that these scientifically unjustified predictions are related to the unique "boom" that all cybernetics has undergone. From a skeptical attitude and even denial of it during its initial stages, public opinion has partially gone to the other extreme. We have made attempts to investigate the human brain by modeling it on a computer for the purpose of creating something resembling an artificial mind, but nothing came of it. This, however, does not mean that the scientific field that has received the not altogether successful name "artificial intelligence" will not become of more importance with every passing year.

[Question] What is the essence of this scientific field?

[Answer] Computers, with their hardware and software and artificial intelligence, are now used in different areas as assistants to man's intellectual activity. Under the conditions of automation of production, there must also be automation of the control of the national economy. Here a certain balance must be observed. Otherwise we will encounter the phenomenon that is called "the information crisis." With the rapid growth of the output list, technological processes and every type of service, the administrative sphere is ceasing to deal with the planning and control processes. This is manifested, for example, in the deficit and nonfulfillment of reciprocal deliveries. Already in the developed countries, one-fourth to one-half of the workers are engaged in processing an endless flow of paper in different institutions. An acute necessity for the automation of the work of office workers, administrators, planners and so on has arisen.

[Question] It has been calculated that the productivity of office workers' labor increases at a rate of about one-fifth that of blue collar workers. How can the effectiveness of office workers' labor be increased?

[Answer] Only through the mass use of computers. Here I would first like to talk about the rapid development of personal computers. They are small in size and simple to use. Their special feature is that with them one can

follow every stage of the computations and make necessary corrections on a timely basis by using a small graph plotter, a display screen and a communication device. They can be connected to considerably more powerful computers as an intelligent terminal and, what is no less important, they can be used to form computer networks. The extensive use of such networks would mean a revolution in the work done by offices, planning offices and institutions.

[Question] How would this look in practice?

[Answer] Let us assume that I need to make some calculation in the field of planning. I make the computations on a computer and find out that for the output of some product, certain equipment and production capacities are needed. I ask the next department what it can give me, and new data light up on my screen then and there. I recalculate, but again I do not get anything. I turn to another department, then to a third one if necessary, and finally I find everything that I need. Quick and easy. In a few minutes or an hour, work that now takes months of correspondence can be done.

[Question] But, as they frequently say, that is not speaking to the subject. Paper is needed.

[Answer] An inquiry is not just a telephone conversation. This is a document that is lit up on a screen and printed on paper. However, not just the papers themselves move about, but their meaning and content, which is also changed: there is no formalism, just the heart of the matter. When one turns to a partner who is in another department or establishment, the outline and text of the inquiry appear on his screen. If some setup does not suit someone, it can be touched up or replaced.

[Question] But what about signatures? Outlines and documents are invalid without them.

[Answer] If you want to sign a document, you need only pronounce your last name clearly over the telephone. The computer prints it and the document takes on legal force. The human voice is unique, just as fingerprints are. If a "sample" of it is retained in a computer's memory, the computer can identify it among millions of other voices. This eliminates "forgery" in signing by another person who, naturally, has a different voice.

[Question] The use of computers everywhere has made the problem of communicating with them an acute one. Right now programmers stand between the user and the computer. These are specially trained people who compile programs in special languages that are understood by the computer. This makes it very difficult for specialists to have access to computers who do not know the secrets of programming. Is anyone doing anything so as to "set up a conversation" with a computer in a natural language; that is, the language in which we speak?

[Answer] That is a very important question. Although at the dawn of computer technology the software and informational support constituted an insignificant share of the cost of the equipment, now the software cost is about 70 percent.

In the Soviet Union, the "POET" information retrieval system has been built and is being used successfully. It is intended for the storage of economic information and is distinguished from similar systems by the absence of intermediaries between itself and those who obtain information from it. It is sufficient to print an inquiry in a natural language, and in 15 seconds a textual answer will appear on a screen. Such systems, with constantly up-dated data, are a remarkable tool for gaining knowledge about the economic and production situation and are of huge importance for planning and controlling production.

[Question] Does this system react to a voice?

[Answer] At the present time a question can be asked of the "POET" system only via text, and it answers with text. The specialists are now working to see that the "POET" system can understand a question asked by a human voice. In this case it is necessary to add phonetic analysis to the morphological, syntactical and semantic analysis of the question. This is an even more complicated problem, and in this area we are taking only the first--yet promising--steps. Systems have been developed that are capable of understanding a very primitive text given by a voice and printing it, but the problem of an automatic stenographer has not yet been solved.

[Question] At the beginning of the 1950's many people assumed that computers could translate from one language to another because of a sharp increase in computation speed and memory capacity. What is the situation now?

[Answer] Everything has turned out to be considerably more complicated than was assumed at first. High operating speed and memory play a role of no little importance, of course. However, the main difficulty is to teach the computer to "understand" the meaning of a phrase. People understand each other excellently, although in their speech they frequently omit, it would seem, needed words. However, the life experience of the participants in a conversation and their perception of voice intonations and gestures make it possible to fill in the missing elements of speech. There is a vast amount of information that is contained in the consciousness in the form of separate bits of information, facts and rules that are ordered there according to meaning and supplemented by the brain's vast associative capabilities. A computer does not have this. This is why the organization of a computer's memory, which gives a good account of itself during the solution of normal problems, proved to be unsuitable for the solution of intellectual problems, in particular translation from one language to another. When it became possible to endow a computer with the capability of extracting and conceiving the meaning of incoming information, automatic translation became a more realistic matter. At Informelektro [probably Information Services and Exhibition Department, Ministry of the Electrical Equipment Industry] and the All-Union Center for Translations, scientific and technical texts are translated by a computer on a sufficiently high level.

[Question] And what about artistic works?

[Answer] This problem is incomparably more difficult. The text of an artistic work is extraordinarily rich in figurative expressions, words that take on

different meanings according to the context, and phrases in which temporal, spatial and causal relationships are hidden. Although special logics that make it possible to determine the meaning of phrases have been developed for the analysis of texts, nevertheless there have been practically no successes in the field of artistic translation, because there everything is more complex than with scientific and technical texts.

[Question] The external world is reflected uniquely and individually in the consciousness of every human being. If we want a computer to do intellectual work, should it not probably have its own unique model of the world surrounding it?

[Answer] The question is how much the model of the world introduced into a computer will correspond to the real world surrounding it. What one can expect from it depends on this. A human being, for example, who has a model of the world can, on the basis of incomplete information coming from this world, form an idea about something. Here I would like to mention one of the areas of artificial intelligence that is now being developed intensively. This is the creation of so-called intelligent robots that are capable of independent action for some period of time. Enthusiasts think that such mechanisms are just about to appear. However, one cannot count on this in the immediate future. The trouble is that in the model of the world of, for example, a transport robot, it is incredibly difficult to take into consideration the constantly changing road situation. Therefore, robots will function in a mode providing for the participation of man for a long time yet.

[Question] But are there not self-teaching robots?

[Answer] To many people their capabilities seem to be exaggerated. In order to learn something, a robot must have a rather complex model of the world. And if this is not present, how can one talk about learning?

[Question] The successes achieved in the development of computers that play chess, compose music, draw pictures and write poetry are widely known. Does this not mean that artificial intelligence already exists in this sense?

[Answer] To one degree or another, all these creative goals are related to searching for and making decisions. When playing chess, a good player thinks about his position 15 moves ahead. In a brief period of time he examines a multitude of variants that are an infinitesimal part of the possible ones, discards the unwanted ones and finds the optimum move. How this process goes on in his head, we do not know. The simple sorting out of the variants one after another would take millions of years. Chess programs for computers capable of playing on the level of masters of the sport have been constructed on the idea of rejecting the unwanted variants. International chess tournaments using such programs are widely known. Special automatic chess players are sold in some countries. For one move, such an automatic unit carries out about 30 million operations per second and examines the variants for only 5 moves ahead. In order to consider 15 moves, the capacities of all the computers existing in the world would not be sufficient. So the capabilities of such programs are still quite limited. Real success will be achieved when the

system of discarding unwanted variants is replaced by a system for generating "good" variants. This will possibly also make it possible to improve the computer "composition" of poetry, music and so on. By the way, then there will also be grounds to call a robot a poet, composer, or artist, as journalists love to do. The modeling of creative processes gave birth to the term "artificial intelligence," but this does not mean that a computer actually has it. The "intelligence" was packed into it by the specialists who developed the program for the solution of some specific problem. The difference between a human being and a computer is that the human does not simply carry out a program stored in his memory, but creates them himself, starting from the goals he has formulated.

When talking about an electronic composer, a zealous correspondent can advisedly imagine to himself an electronic journalist. I think we do not need any special evidence to understand the absurd attempts to replace a special correspondent, the head of a department, or an editor with computers...Here there is room for humor only, and not for serious science.

COPYRIGHT: Izdatel'stvo "Pravda", "Zhurnalist", 1984

11746

CSO: 1863/231

## LENINGRAD COMPUTER REPAIR FIRMS FAIL TO GIVE NEEDED SERVICES

Leningrad LENINGRADSKAYA PRAVDA in Russian 22 May 85 p 2

[Article by V. Tveritina and V. Chichin]

[Abstract] The lengthy article reports on problems of enterprises and organizations of Leningrad in getting their computers repaired. It is generally critical of two computer repair and maintenance associations, "EVMservis" and "SeverEVMkompleks," which were created to provide centralized, full-service repairs for computer owners that contract with them.

In general, it is said that clients of the repair firms complain about long waits to get service. Often they are told that needed spare parts are not on hand. At other times, repairmen say they aren't able to repair a computer because they aren't familiar with the model. A new computer at the All-Union Geological Institute has been idle for a year for this reason. L. I. Min'kov, chief engineer of "EVMservis," acknowledged that many clients are dissatisfied, and gave the excuse that nearly one-third of the firm's staff at any time is committed to advanced-training courses to keep up with new technology. But the authors of the article found that the firm has been slow to introduce automated equipment for fast diagnosis of computer malfunctions. V. I. Novitskiy, head of the department of computer technology of the Main Special Design Bureau of Thermophysical Instrument Building, expressed the suspicion that the repair firm benefits from frequent computer breakdowns, since the client pays for each service and the contract does not provide for compensation to the client for poor repair work.

As a result of the situation, it is said that organizations which have a large number of computers, such as the association "Lensistemotekhnika" (systems technology), which calculates payroles for enterprises, and the research-and-production association "Lenelektronmash," are employing full-time repair staffs, shunning the practice of outside contracting.

It is also pointed out that the two repair firms are specialized for classes of computers, one for mainframe computers and the other for smaller computers. But for mini- and microcomputers, which are becoming the most numerous class, there is no specialized repair firm in Leningrad. Owners of these computers must go to Krasnodar for repair services.

In conclusion, the authors say that Leningrad needs a single center for full-service maintenance of all classes of computers. It could incorporate the two existing associations plus a number of smaller enterprises. This center would also be a computer sales center advising prospective owners on the type of machines that are best suited for their needs.

CSO: 1863/353  
FTD/SNAP

## NEED FOR COMPUTERS, AUTOMATION IN PRODUCTION CITED

Moscow SOTSIALISTICHESKAYA INDUSTRIYA in Russian 2 Apr 85 p 2

[Article by Academician I. Glebov, chairman of the Leningrad USSR AN Scientific Center Presidium: "Integrating Science And Production"]

[Text] The goal of "Intensification-90" is to radically change the economy of Leningrad and the oblast through intense development. To do this, branch and regional interests in it are coordinated in such a way that they are focused on finding and using the reserves of the enormous region that has a significant production and scientific-technical potential at its disposal.

A large part of these reserves lie in the area of scientific-technical progress. More and more often today during production reconstruction and reequipping enterprises have to resolve tasks that are similar in character. For example, these tasks include questions of using computer equipment, comprehensive automation and production mechanization and the installation of progressive processes such as powder metallurgy and laser, plasma, electrophysical and electrochemical methods of processing. One doesn't have to use a computer to come to the conclusion that by combining the efforts of collectives, if only in the area of research and development, and by setting up an exchange of their experiences, they can significantly reduce both the time for resolving these tasks and also expenditures.

And there are other reserves that are being reviewed at a regional level. We know that in many instances when enterprises and organizations in related ministries are solving a common economic task, they operate independently of one another, although figuratively speaking they are located in the same street. The desire to overcome this departmental separation is the basis for the organizational work which the Leningrad Party Organization has been carrying on for a number of successive years. Its results have shown up most visibly in the example of collectives working in the energetics field.

On the eve of the present five-year plan these collectives came out with an initiative to increase their contribution toward solving the energy problem. As a result, Leningrad's comprehensive "Increase The Efficiency of the Country's Fuel-Energy Complex" program was developed. This combined the efforts of more than 200 participants, including academic and specialized institutes, VUZ's, associations and factories and construction organizations.

The concept of this program was not to aggregate the planned tasks of ministries and carry them out or exceed the plan. From the very beginning the plan was aimed at attaining serious transformations in the activity of Leningrad enterprises and organizations, and at affecting the necessary changes in both nomenclature and in production volume.

This required that new plants be developed and existing one be reconstructed, specializations not only in the production sphere but also in science be expanded and the efforts by collectives with different subordinations be directed to reach a common end result. These issues required that there be careful coordination and agreement with the appropriate ministries and departments, among regional and centralized planning organizations. This experience played an important role: /During the development of the "Intensification-90" program, all inter-branch interaction issues were resolved within the framework of the existing economic mechanism by very detailed coordination of plans at a regional level/ [boldface].

True, it was not easy to do this, as the program encompassed practically all the economic divisions in Leningrad and the oblast. More than 330 enterprises and organizations subordinated to 99 ministries and departments took part in its development. These numbers make it easy to picture the scale of agreement. Moreover, the fundamental tenants of the program were reviewed at the USSR Academy of Science Presidium, in the USSR Goskomitet [State Committee] on Science and Technology, the union Gosplan, the RSFSR Council of Ministers and the republic Gosplan. And only after this were the plan's programs included in the state economic plan for 1985 and entered into the design of future five-year plans.

The following information can be used to show the results of this preparatory work: the plan called also for an 100,000-man reduction in industrial-production personnel as compared to the original ministry and departmental drafts. And by what means will these results be achieved? It is difficult to give a well-defined answer to this. The program intends to carry out 2,600 large-scale measures in very different plans, but there is still an over-all concept. /The program's basic idea is to speed up scientific-technical progress in all economic areas with the primary direction of the work being the comprehensive introduction of computer technology and automation/ [boldface].

At first glance this approach could seem extraordinarily narrow, but it was prompted by the specific industry of Leningrad and the oblast. Many of our enterprises are distinguished by their small scale and even unit production character. For example, even within the confines of a single series, often every dry-cargo ship that is launched into the water of each successive gas turbine has the mark of significant improvements. This is dictated by the collectives' dedication to constantly improving the quality and technical level of the products they manufacture.

In actuality, this desire is manifested in the large volume of research and design work. Reducing their time-frame and raising their levels while limiting the number of personnel can only be done by using modern technology based on EVM [computers], an automated scientific research

system (ASNI), automated design (SAPR) and automated work places for specialists (ARM). Electronics allows us to immediately turn over delivery to the production manager. His labor efficiency can be increased by using automated production engineering systems (ASPP). Thus the control programs which can be transmitted to rigs and machines using communications lines with ChPU [numerical program control] and automated systems must be some "products" of these systems.

Of course, production must have available the capabilities to rapidly reorganize for the output of new or modified products. A flexible production system (GPS) which is again controlled by a computer answers this need to the fullest. Thus computers do more than just reduce the "research--production" cycle. They also become the connecting thread which allows us to bring all the links together into an comprehensive, integrated production complex (IPK). As calculations are showing, such complexes are twice as efficient as automating individual processes. Therefore /the trend toward developing integrated production complexes is one of the decisive scientific concepts in the "Intensification-90" program/ [boldface].

At the same time the water on automation and EVM computers make sense in another way. One can certainly recall many instances where installing the most modern ASU [automated control system] did not bring the desired results only because the technical level and organization of production itself kept one from hoping for better. The attempts to automate control of unproductive, old machines, equipment and technological processes began very deplorably. From this point of view, /it is precisely computers, robots, and flexible systems, with the appropriate demand for efficiency, that must become the catalyst for technical progress and make the search for more improved technical, organizational and administrative decisions necessary/ [boldface].

Most of all, this search must provide a significant increase in the return from existing basic capital. Up to 80 percent of capital investment is planned for reconstruction and technical reequipping. Automation and especially flexible production systems must create conditions for the rapid assimilation of new products, an increase in the equipment shift-work coefficient, a reduction in the processing cycle and an increase in the level of labor productivity. At the same time progressive, resource-economizing technology and group methods of processing will be used extensively and production specialization and cooperation will be expanded.

12511  
CSO: 1863/304

NEED FOR MICROCALCULATORS, NEW NEWSPAPER COLUMN GIVEN

Moscow SOTSIALISTICHESKAYA INDUSTRIYA in Russian 16 May 85 p 4

[Article by I. Danilov, Candidate of Technical Sciences: "The Computer In the Work Place"]

[Text] Today we are starting a new section. The newspaper will use this section to write about types and capabilities of microcalculators and personal computers, about tasks given them, and about how to turn them into faithful aids at work, how to develop simple and easy programs to solve engineering, design and production tasks.

I don't know if man underwent a painful transition from the goose-quill pen to the steel pen, but I remember when the ballpoint pen first appeared in school. How hot the controversy was! "Don't allow the ballpoint pen! They will ruin children's handwriting" teachers claimed. Today people write with steel pens as well as people did before they appeared.

Something like that is happening now with computer technology, or more specifically, with its most widespread representatives, microcalculators. These small computers are being produced today by the hundreds of thousands every year and are invading our lives in a commanding manner. They are already a social phenomenon and must therefore be taken very seriously.

"Microcalculators are weaning children from calculating and they are becoming blind appendages to computers," some school teachers fear. Even some VUZ instructors feel that computers are depriving students of a feel for numbers and of the ability to evaluate results.

Some of the mistrust for microcalculators that established specialists have is explained in another way. What if the computer suddenly makes a mistake? How do you check it? It is better to calculate on paper or on a slide rule. At least here everything is visible. And finally there is one more category of angry people--professional programmers. "Why campaign for microcalculators when personal computers already exist? This is like telling a motorist to change to a bicycle seat."

Well, let's look at the facts. Specialists know that school children who use a microcalculator at home solve problems better than their contemporaries who never use them. True, these students learned to calculate correctly before they began to use a microcalculator.

Now, about errors. The reason for mistrust here is purely psychological. Specialists who have not worked with computers are used to computing with the numbers right in front of their eyes, either on paper or on a slide rule. When the numbers "disappear" into the computer's memory, they begin to have doubts about what is going on. And the use of mechanical calculators promoted this mistrust. There was an abundance of gear wheels in the arithmometers and when electrical connector relays wore out, the calculators began to "lie". In modern computers, only electrons move and they do not grow old or wear out. Therefore, if there are errors when calculating with a microcalculator, they are the fault of the individual who pushed the wrong keys or pushed them in the wrong order. And it is afterall possible to make mistakes through inattention when multiplying with columns.

Objections from professionals are more serious. Actually, the capabilities of personal computers, whose production our industry is beginning to assimilate, is significantly greater than those of a microcalculator. But we will be realistic. First, the volume of computer production is still small and they are still expensive. Second, you need a specially equipped work space for them, whereas you can use a microcalculator which is a portable computer, everywhere--in the workshop, on the construction site and in the field.

As for those people whose inertia is holding them at the abacus, it is worth noting that microcalculators are super-fast calculators and the most accurate slide rules, and they are significantly easier to use. And at the same time they can compile all possible mathematical tables and list these tables in only seconds.

The extensive use of microcalculators in schools frees time for more comprehensive instruction in mathematics. Microcalculators allow an engineer to evaluate technical ideas faster and more accurately. And a lathe operator can correctly select his instrument and work characteristics when turning various components. In general, it is difficult to name an area where a computer would be superfluous. For example, a small computer in the hands of a salesman would make it easier for him to calculate the cost of a purchase and save you time.

Interaction with a computer not only saves you from the routine part of mental activity, the manual calculations. It also trains its owner in neatness, attentiveness and accuracy and can develop a certain programmer style of thinking, the ability to formulate a problem clearly and find an optimal solution for it.

And there is more. The microcalculator is opening the door a crack to the limitless world of computers, a world where we will have to live in the near future. Indeed, even today each of us is having a difficult time digesting

the avalanche of information with which life is plastering us. Tomorrow it will be impossible to do this without the help of a computer. Therefore a man who lacks this second literacy, programming, will be in the same position of a man today who cannot read.

It is easy to learn how to calculate on a computer, but it is more difficult to master one. And our new rubric has been started to help the million-strong army of microcalculator owners do this.

12511  
CSO: 1863/326

## DIFFICULTIES IN FACTORY AUTOMATION TOLD

Moscow PRAVDA in Russian 7 May 85 p 3

[Article by P. Derunov, General director of Yaroslav Oblast's Andropovskiy Industrial Motor Building Association and Hero of Socialist Labor: "Rebuilding On The March"]

[Text] In carrying out the plan for technical re-equipping the collective at the Andropovskiy Industrial Motor Building Association has doubled its volume of production and labor productivity in the last nine years. How were they able to speed up the tempo of growth?

Intensification has radically changed production. More than half of the workers in the primary shops operate automated equipment and the enterprise is now starting a qualitatively new stage of re-equipping. In the end, we have to develop an automated factory complete with automated equipment complexes and computer technology both in production itself and in its training and management.

Recently GPS [flexible production system] has become "stylish". Even mass-production enterprises have declared their intention to use this system. When they do this, they do not always consider that flexible automation is a complicated, expensive, science-intensive affair. And introducing it into all types of production simultaneously diffuses the effort and wastes valuable time. We feel that GPS should be developed at the first stage in series production.

Flexible systems must only be developed on a foundation of reliable and highly productive equipment. The reliability of machines with ChPU [numerical program control] have special significance, for if any element in these machines goes out, it can cause downtime in the whole complex.

Production automation based on the contemporary level of technology demands a fundamentally new approach to control. We must monitor the entire preparation process and not the individual part. We in the association have given up having the OTK [Department of Technical Control] monitor each individual operation and have increased foreman, brigadier and worker responsibility for the quality and stability of the product preparation process. All of their parameters must be controlled both as special units of production equipment and as methods for monitoring automation that are built into GPS.

In order to realize one of the advantages of flexible systems, e.g. speed of gearing up for new types of products, we must also introduce automated design systems (SAPR). The primary problem in developing them is equipping the enterprise with modern computer technology and the appropriate peripheral equipment and software. In our opinion, it is necessary to increase production of this equipment along with its unified software in the very near future.

The problems of developing advanced GPS and SAPR equipment are so complicated and inter-related that they require basic research and the union of industry and modern science. We are expanding ties not only with the leading specialized institutes, but also with the USSR AN [Academy of Sciences] scientific centers.

The level of tool and fixture quality, and increasing their production have major significance in the effective operation of flexible systems. It is enough to say that every component that is manufactured must have as a minimum three sets of tools, one in the machine store, one in the automated warehouse and one in reborning or repair. The tool must meet contemporary requirements. It must be very reliable as well as accurate and must guarantee an increase in cutting and delivery speed, in the number of operations it carries out and in the inventory of materials that are processed.

In considering this, we feel that the creation and introduction of flexible automation must follow the reconstruction and development of tool production and the boosting of its organizational and technical foundation to a new qualitative level. We feel that large associations must encompass tool factories with very productive and precise equipment. It would also be advisable for the USSR Goskomtrud [State Committee for Labor and Social Problems] to provide for such factories in the standard management structures of associations.

We feel that it is unrealistic to focus only on flexible systems when carrying out technical production re-equipping. You can't go directly from manual labor into the world of total automation. You have to first give production some mechanized means and programmed machinery, then develop modules and sections controlled by computers on top of the initial automation. These then become cells for future GPS. In order to resolve the problems of technical re-equipping at a high level and in a compressed time-frame, in my opinion we must produce mechanization and automation equipment more quickly by setting up specialized enterprises. In this way every factory can acquire this equipment. Because our association plans to install 150 robots in 23 robot-equipped complexes in the next five-year plan, we are specifically interested in purchasing robots, as we have not solved the problems of making them ourselves.

Accelerating technical progress requires that we realign worker, engineer-technician and managing cadre consciousness, increase the growth of their qualifications and increase their output. The interests of intensifying the national economy require that we create conditions for enterprises to show initiative and enterprise. They must specifically develop ties among

themselves. This form of cooperation is advantageous not only for direct partners but also for the national economy as a whole. For example, an association manufactured a pneumatic tractor for one of the metallurgical factories. This tractor was used to clean uncooled furnace flues which reduced their down-time and allowed additional thousands of tons of metal to be produced annually. Some of this metal then came to us and was used to increase production.

We have similar agreements with the Ivanovskiy Machine Tool Manufacturing Association. We made 15 tables for each of the association's machine manufacturing centers which they made for us above and beyond their plan. Because of this we got several manufacturing centers. However the future of this cooperation is unclear, for despite the support from our ministries, planning agencies are hindering the development of our association's direct ties with the Ivanovskiy Machine Tool Manufacturing Association.

The increased practice of numerous controlling agencies and inspectors using the extensive rights that they have not for the benefit, but to the detriment of business, stopping the work of many collectives instead of punishing specific guilty parties, is having a negative affect on the successes of enterprises. For example, recently when we slightly pushed back the time for putting in drainage canals because of a lack of financing, the reservoir inspector who was dissatisfied with this used the bank to hold up all other construction-assembly work.

The practice of pulling enterprise workers into construction and other jobs is severely impeding the course toward intensifying production. Engineer-technical workers are being sent to construction sites more and more often. This distraction reduces their creative output, disrupts their planned conduct of work and reduces the tempo of technical progress in the enterprise. And the harm is two-fold, as it affects both the factory and construction organization. The people pulled away from their work have their spirits dampened and so do the construction workers. Moreover, if you succeed in construction not because of numbers, but because of a well thought-out work organization, you can get good results. Even our experience supports this. We have been doing our own construction for many years and recently we built a 540-apartment 16-story brick building in a year instead of the normative two-and-a-half years.

The sequence for planning outlays for reconstruction and technical re-equipping must also be improved. At the present time the need to replace machines is taken as the starting point for calculating allocations for upgrading basic capital. In reality, the work position, including the equipment, the area for housing it, service, and the tools and organizational equipment, is the basic unit for improvement. Therefore the volume of centralized capital investment for technical re-equipping must be planned according to the need for upgrading an established number of work positions. This can be done on the basis of cost normatives that provide the necessary growth of labor productivity.

We feel that the method for calculating the effectiveness of putting new equipment into production must also be changed. It is not complete, for it takes into account only the purely production effect. You also have to take into account the social effect from reducing the number of personnel as a result of modernization. Many industrial enterprises have a developed social-domestic sphere and they spend a significant amount on its support.

Having developed a program for automation, the association established the capability of providing for the high tempo of development for the Twelfth Five-Year Plan. Production volume and labor productivity will grow by more than 40 percent. The outlay for automation, including the social affect associated with reducing the number of workers by 800-1000 men in the five-year plan, will pay for itself in 2.5 years.

Solving the issues raised here would facilitate the acceleration of the technical process in the country's machine building enterprises.

12511  
CSO: 1863/326

## COMPUTER IN A BOX

Moscow MOSKOVSKAYA PRAVDA in Russian 9 Feb 85 p 2

IVANOV, A., Chief, Computer Center, City Statistical Administration

[Abstract] Twice per year, all organizations and enterprises in the capital which operate computer devices present reports to the City Statistical Administration. Recent reports indicate that 7.6% of all computers are totally unused, many because of startup difficulties. The computer room, power supply and other associated devices are frequently not ready when the computer is delivered. This article describes the damage to the national economy due to useless capital investment, unutilized information and un-produced reports. More serious than unpacked boxes is the damage done by enterprises that are unprepared to use the computers allocated to them. The author states that specialists who will serve the computer should be hired or trained at least one year before the scheduled delivery date of the machine to reduce the number of computers in the city still in their boxes.  
[178-6508]

## STANDARD DESIGN - THE BASIS OF CREATION OF THE AUTOMATED SYSTEM OF FINANCIAL CALCULATIONS OF THE RUSSIAN FEDERATION

Moscow FINANCY SSSR in Russian No 4, Apr 85 pp 53-58

FILIMONOV, B. I., Director, Information and Computer Center, RSFSR Finance Ministry, KOLESNIK, A. P., Deputy Director, and KOVALEV, A. F., Department Chief

[Abstract] In the Russian Federation at the oblast' level, practical work on the creation of the automated system for financial calculations (ASFR) is underway at nine computer centers throughout the federation. The ASFR includes some six hundred tasks. Standard design has been selected as a method to reduce the cost and accelerate the development of the ASFR in the federation. In order to define the necessary conditions and suitable objects for standard design, the article discusses how an ASFR task is run on a computer. The end result of development of a standard ASFR task is a program. The program of a typical task must run on all computers used by the

financial organs to perform ASFR jobs, meaning that the computers must all be compatible at the software level. Since this is not the case, several versions of programs must be developed. The conditions required to assure standardization of developments at the information computer center have been given great attention from the very start. In 1981, the RSFSR Ministry of Finance issued an order assigning to the information computer center the functions of creation and introduction of branch classifiers. A conference held in Gor'kiy in 1982 discussed problems of information support, particularly classification and coding of indices. Only the information computer center has the right to change the system of branch classifiers. The system of branch classifiers is the language used by computer centers at the oblast' level to communicate with each other and with higher organizations. This will facilitate expansion of successful experiments such as that conducted in Gor'kiy in 1982 in which automated and manual accounting operations were merged and jointly operated.

[317-6508]

HARDWARE

UDC 621.382

PROSPECTS FOR CREATION OF OPTICAL DIGITAL HIGH-PERFORMANCE COMPUTERS

Novosibirsk AVTOMETRIYA in Russian No 1, Jan-Feb 85 (manuscript received 10 July 84) pp 114-126

[Article by V.M. Yegorov and E.G. Kostsov]

[Text] The main driving force of the development of modern microelectronics has always been the requirements of computer technology and in recent decades microelectronics has attained important successes. It is sufficient to mention the following figures: the value for energy dissipated by an element during the execution of an element logic operation fell to  $10^{-12} - 10^{-13}$  J [1], switching time for individual circuits fell to 30 ps for n-MOS (circuits with submicron channels [2]) and to 12.8 ps for 1.1  $\mu$ m HEMT (high electron mobility transistor) circuits given 77K [3] while the level of integration approached  $10^9$  elements/chip.

The productivity of large computers simultaneously increased to  $10^9$  operations/s essentially because of an increase in the response speed of the discrete elements and reductions in the length of the clock period which for modern large-scale machines has a value of  $\sim 10$  ns [4] and, according to forecasts, will fall by 1990 to 1 - 2.5 ns.

However requirements for further qualitative improvements in output exceed the actually attained results since requirements increase at faster rates. For example, the large-scale problems requiring the creation of "supercomputers" include a series of economic problems and problems from the areas of space research, mathematical physics, etc..

It appears that the switching times mentioned above for discrete elements are a basis for the supposition that there will be further sharp increases in computer productivity. However, in spite of the continuing progress of the element base, there is at the present time a situation which is qualitatively different from that which was characteristic of the preceding period of development of computer technology. The change in the situation is due to the following reasons.

1. In the contemporary VLSI the area occupied by the conductors is almost equal to the area of the chip [5], the overall length of the conductors is more than  $10^3$  times greater than the linear dimensions of the chip [6], the energy dissipated in charging the conductors already amounts to 70-75% of all energy losses and the limiting value for linear capacitance is practically reached for the conductors ( $10^{-11}$  F/m) [1]. At the same time, it is characteristic for modern logic circuits that the number of connections is significantly higher in comparison with the number of function elements, i.e., the structure and reliability of large computers are determined by the structure and reliability of the connections while with the in-

crease in the packing density and the rise in the response speed of the elements the role of connections increases along with the level of transient noise.

In addition, in the creation of VLSI it is practically impossible to realize connections only on a single plane and multiple assemblies are necessary with the number of levels at least greater than three [7].

Even more complex problems arise in the interchange of information between substrates. Rent's rule [8,9] indicating the proportionality of the number of pins from the substrate to the number of logic elements located in it is empirically satisfied for computers. A drop in the number of pins leads to a drop in the technical characteristics of the unit [10]. At the same time, as chip dimensions increase their perimeters (allowing only a certain number of external pins) increase linearly while the area used under the element increases by the square law while the number of connections grows even more rapidly.

Even today, the connections between chips reduces the clock frequency practically by an order of magnitude. In the future, with the reduction in the clock period, communications between chips will play an increasingly important role which will result in a limitation on the clock rate of unit operations. In addition, the operation for the creation of connections between chips by means of conductors is clearly not compatible with the technology for the creation of the IC themselves and causes both a large percentage of the waste and a relatively low level of reliability for the units.

In [1] it is correctly noted that the complexity of the units limits the level of integration of the chips to levels which are significantly lower than might be allowable within the limits of semiconductor technology. In particular, estimates show that in logic units, the limiting level of integration is three orders less than in memory matrices [11]. The reason is the more complex character of the connections.

2. The qualitative change in the situation is also linked to another factor which also does not result from the specific features of the physical phenomenon which is the basis for the functioning of the logic elements. The fact is that for large units which have sufficiently extensive geometric dimensions, further reduction in the clock period has no meaning because of limitations on signal propagation time.

The maximum extension of the communication channel in the unit is equal to  $L = c\psi t$  ( $c$  is the speed of light in vacuum,  $t$  is the clock period,  $\psi$  is the coefficient taking into account phase shift and lag in the linear transmission of the signal). Supposing that  $t = 10^{-9}$  s,  $\psi = 0.1$  (this value can be increased by a factor of several times and, in addition, it falls with increases in the level of integration) and we have  $L < 3 \cdot 10^{-2}$  m. For contemporary technology for the exchange of information between chips it is characteristic that only some conductors in large computers occupy a volume larger than  $L^3$ .

Thus, the main reserve for increasing the productivity of computers of the previous period of computer technology development (increasing response speed) is practically exhausted.

In recent years, the main directions for increasing computer productivity have become more clearly marked and consist of the parallel development of data processing inclusive of element operations (deepest level of the parallel development process) and decentralization of the memory, control and data transformation functions.

This type of parallel process is naturally obtainable by corresponding progress in the production of microcircuits and, first of all, in the fabrication of connections.

However, as was mentioned above, contemporary microelectronic technology cannot produce significant progress as concerns the connection problem. Therefore, the time has come for serious consideration of the question of establishing a new element base for computer technology and only the solution of the connection problem and the corresponding sharp increase in productivity can lead to the appearance of an economically viable new technology for the fabrication of electronic circuits. It is also clear that the units to be developed should allow integration with an element density which is commensurate with the density of elements in modern VLSI while the technology for fabricating these units should be based on the achievements of modern microelectronics. Naturally this type of approach excludes the use of discrete macroscopic elements such as, for example, lenses, prisms and mirrors.

The creation of a new element must be guided by an aggregate of requirements concerning the logic elements [12] and the observation of the principle of homogeneity when the functioning of the element is based on a minimum number of physical effects (not more than two) as in modern logical structures. However the main requirement is the solution of the problem of the connections. At this level, for example, the application of elements based on superconductivity necessarily leads to the same problems for the organization of connections which occur at the present time.

From general physical considerations it follows that the introduction into the structure of "three-dimensional" links is easier to carry out by means of optical link channels. But the optical link channel differs from the electron conductor because of several drawbacks of which the following can be mentioned: variation in the direction of signal propagation is complex and not efficient, the minimal dimensions (diameter) of the optical channels are limited by the wave nature of light. The absence of direct interaction of light fluxes leads to a need for energy conversion with corresponding energy losses. At the same time, the presence of a solid energy converter gives an immediate answer to the question as to the parameters and limiting dimensions of the elements, their response speed and arrangement density: they cannot exceed the corresponding parameters which are characteristic for modern microelectronics.

Therefore, application of an optical link channel in computer technology is expedient only in the case that a concept will be developed for effective use of the advantages of a three-dimensional link structure and simultaneous exchange of information between a set of elements located in different substrates. We note that the application of optical fiber elements [13] in the creation of large computers does not solve the connection problems. On the contrary, it adds a series of specific fabrication tasks to these problems.

At the present time, a set of elements are known in which logic signals are transmitted by means of light fluxes. The most important and serious developments at

this level are in the area of classical optoelectronics (the base element is the optron) [14]. This direction, however, did not have a significant effect on the development of computer technology mainly because of such physical limitations as the double conversion of energy (electrical signal to optical signal and the reverse) which causes low element quality and the need to compare the different characteristics of each light source-photodetector pair taking into account the source directional pattern. In addition, the overall number of individual directional light sources fabricated for the integrated application has to be commensurate with the number of logic elements ( $10^3 - 10^4$ ) while the light source, as is known, is one of the least reliable elements and is distinguished by its low efficiency.

Logic elements are also known which are based on semiconductor laser light sources using the Koester effect [15]. The interest in these is due to the fact that by using them it is possible to create memory elements with a switching time of  $10^{-11}$  s and all switching and the main logic operations are carried out in the optical branch of the circuit. When the drawbacks of this kind of unit are mentioned these are usually the need to use powerful control circuits ( $10^{-2}$  J/bit) and the low reliability of the lasers. However the main drawback of these elements is elsewhere, i.e., there is still the unsolved problem of the connections between the substrates. In addition, the technology for fabricating logic circuits using laser switches does not give definite basic advantages in comparison with VLSI technology and, in fact, the reverse is true; and it adds a series of problems of its own which are quite complex.

In [16] a logic element is described which consists of a Fabry-Perot interferometer between whose mirrors a narrow-zone semiconductor (for example InSb) is located which has the ability to nonlinearly vary the coefficient of refraction  $n$  under the effects of light flux starting from a certain value of flux intensity. We will indicate the drawbacks of this class of elements which, in our view, are such that they cannot be considered for application in the design of large-scale computers.

Variation in the value  $n$  up to output in the saturation region as a result of light flux is possible only when it is of sufficiently high intensity which causes low element quality. As a result of the fact that the degree of nonlinearity of  $n$  increases inversely to the square of the width of the forbidden zone of the semiconductor, in creating the element it is effective to use only materials with a zone width of 0.2 eV or even less and the wavelength  $\lambda$  of the light flux should be 5  $\mu\text{m}$  or more. This fact together with the diffraction phenomenon shows the impossibility of creating elements with element density commensurate with the element density in VLSI.

The light interference principle requires high precision in maintaining the parameters of each of the elements which, when the latter are mass produced, is a more complex task (by two orders for the precision of reproducibility of the layer thickness) than occurs in the fabrication of modern logic circuits.

For the element under consideration, it is characteristic that the locations of the input and output are superimposed and it is not clear how to design links between the set of elements located on different substrates. When mirrors are used, the distance  $h$  between the substrates becomes commensurate with the dimensions of the substrates which leads to an increase in the diameter of the light beam to the value  $\sqrt{\lambda h}$  and to the exclusion of the possibility of integrated use of the unit.

In order to lower the noise level it is recommended that the unit be deeply cooled. If this is not done, the element has a transfer characteristic giving unattenuated signal propagation sequentially in the switched on logic elements of the circuit.

The discussion given above also holds, to a significant extent, to other similar elements which are based on the interference principle [17, 18].

We had previously [19] developed physical principles for the design of a base logic element which is free from the drawbacks described above. Its functioning is based on the modulation of the light flux by an electric field. In this case, only one energy conversion is used (light to electric signal) and there is a common light source for the set of elements. The analysis given in [20] of the physical effects which can be used as a functional basis for the element showed that the optimal variant is as follows: electrooptic light modulator, photoelectric energy converter and electrostatic energy storage element. Application of magneto- and acoustooptical light modulators is limited by high energy expenditures and by the complexity of direct conversion of (photomagnetic or photoacoustic) energy.

Fig. 1, a shows the circuit of the optical logic element base. It consists of the modulator M, energy converter F and energy storage device, for example capacitor C for dynamic circuits or resistor R for potential circuits.

The state of the element is determined by the intensity  $I_{out}$  of the light flux at the output of the light modulator (or the potential difference on its electrodes) which is unequivocally linked to the intensity  $I_{in}$  of the light signal entering the input of the photoelectric energy converter. The function of the logic element is to control the intensity of the light flux transmitted by the light modulator by means of the light flux entering the element.

Fig. 1, b-h show potential circuits for optical elements realizing the main logic functions: b. repeater; c. inverter; d. AND element; e. AND-NOT element; f. OR element; g. OR-NOT element; h. mod 2 summation element; i.  $X_1 \rightarrow X_2$ , implication element. Here it is supposed that the modulator transmits light if the potential difference over its electrodes has the high value (1) and does not transmit light if the difference has the low value (0).

Dynamic circuits are also of great interest since together with logic elements they also make it possible to realize memory functions. We will consider the operating principles of dynamic circuits for the example of a memory cell (Fig. 2). We will suppose that the unit of information is written in element 1 and the switch K1 is closed on the grounded contact 2.

Step 1. Deletion of information on element 2. Switch 2 closes on contact 1 of power source V and capacitor C2 is charged.

Step 2. Reading of information in inverse code from element 1 by means of light signal  $I_{read}$ . The switches K1, K2 are in position 2 (grounded) and the output signal from M1 is the input write signal for element 2.

Step 3. Deletion of information in element 1. Switch K1 is in position 1 and capacitor C1 charges.

Step 4. Reading of information in forward code from element 2 by means of light signal I<sub>read</sub>. Switches K1, K2 are grounded. The output signal from modulator 2 is the input for element 1.

As a result of the execution of steps 1-4 and repetition of information cycles in the cell, it is possible to store information as long as is required.

In electronic computers, parallel algorithms are realized by means of parallel electric circuits which link discrete elements and structures. It is clear that the number of circuits, gate and switch elements which regulate the propagation of the electric signal in the required direction and supply the necessary functional flexibility and effectiveness increases with the rise in the level of parallelization of the computing process and the universality of the device. The described element base makes it possible to create functionally flexible computers with a high level of productivity without increasing the number of circuits and elements by replacing the electric links by optical links.

The design of the computer will have the smallest number of elements and electrical connections if it satisfies the following principles: 1) parallel electric circuits are used for supplying the elements; 2) parallel functional links are realized by optical channels; 3) functional links between structures and discrete elements realized by electric circuits are in series.

In order to illustrate the possibilities of the indicated element base we will consider the design and operation of the matrix processor element which satisfies the above principles.

The element has six cells (Fig. 3): four inverters containing photoelectric converters (photodetectors) F1-F4, light modulators M12-M15, capacitors C1-C4 and two functional converter cells: F5-F8, M16-M18, C5-C7 and F9-F11, M19-M22, C8-C11 which are also capable of storing information in dynamic mode. The electric inputs F1-F11 are connected to the switching contacts K1-K6 for switching to the power sources V1-V6 or to the low potential bus.

When neighboring elements are connected to the element, the electrical inputs of all the photodetectors of the corresponding cells are connected in parallel with the contacts of the corresponding switches K1-K6. The functional converter has contacts 23-30 for connection with the contacts of neighboring elements of the matrix to the left, right, below and above. The functional power of the element is supplied by optical links between cells. They are (Fig. 4) such that any two cells have interconnected optical inputs and outputs and the first and third inverters are optically linked to the second and fourth inverters of the neighboring matrix element to the left, whose components are marked "L", in the center of the block, while the second and fourth are linked to the first and third inverters of the neighboring matrix element to the right, whose components are marked "P". In addition, M12, M14 and M13, M15 have a connective optical link and all the light modulators of one functional converter cell are optically linked to the corresponding photodetectors of the other. In order to do this, the photodetectors F1-F14 have six inputs each, F5, F6, F9 have two each, F7, F8 have five each, F10, F11 have four optical inputs and each photodetector also has an input for receiving a signal from the external light source. The light modulators M12-M15 have six optical channels each; M16, M19, M20 have one each and M17, M18, M21, M22 have

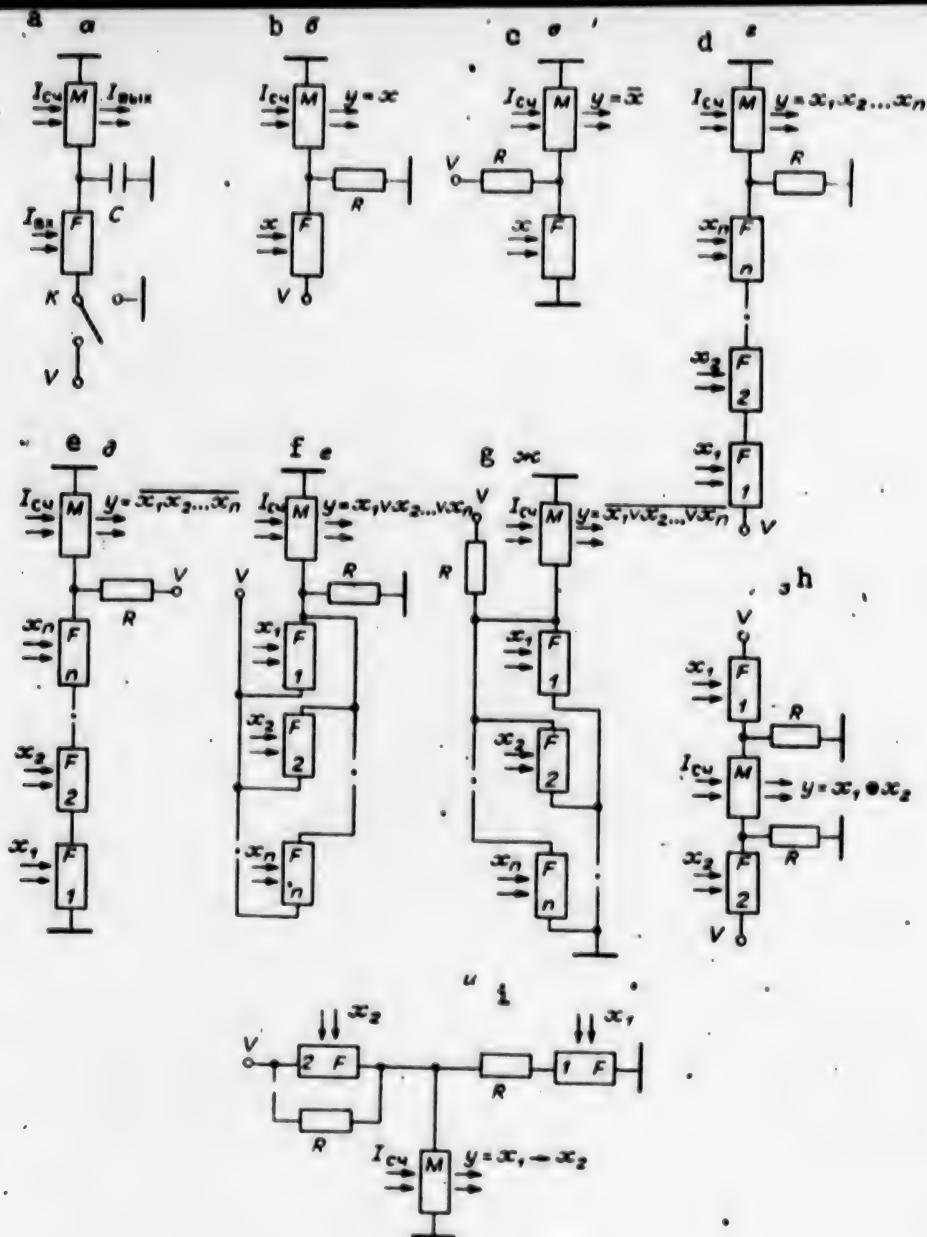


Figure 1

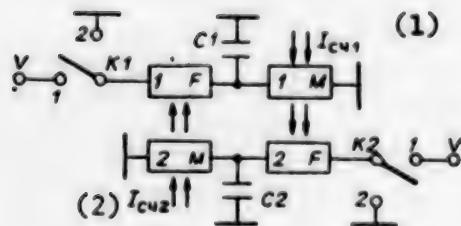


Figure 2

Key:

1.  $I_{\text{read } 1}$

2.  $I_{\text{read } 2}$

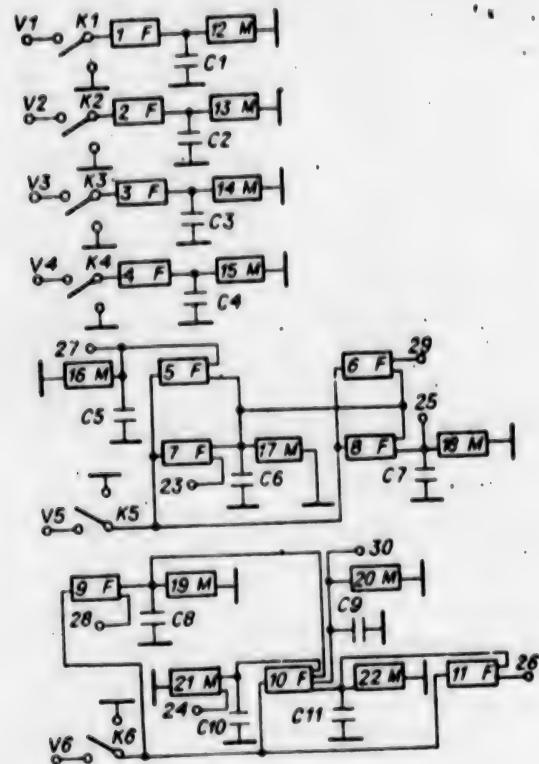


Figure 3

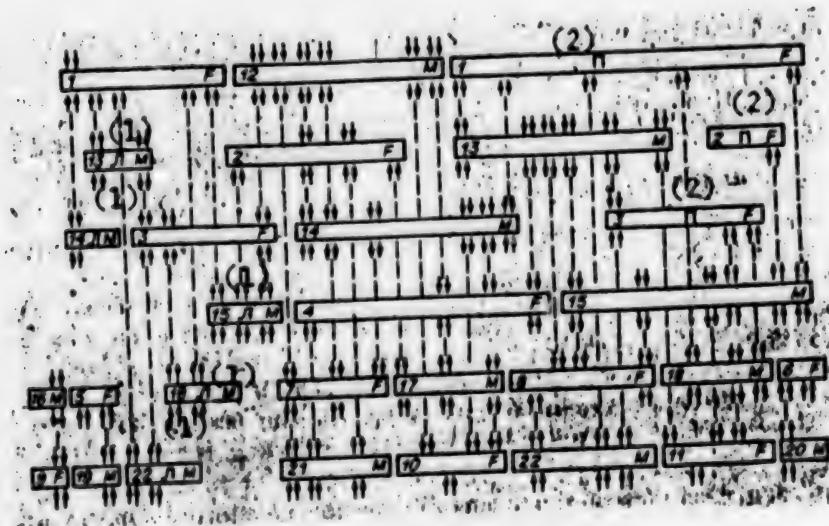


Figure 4

**Key:**

1. L

2. P

three optical channels each.

In dynamic mode, the corresponding inverters form dynamic memory cells with shift structure and the functional converter can store a unit of information with a shift in each of the four matrix directions and without such a shift. For the transmission of the optical signals  $x_1 - x_7$ ,  $a_1 - a_7$ ,  $b_1 - b_7$ ,  $c_1 - c_7$ ,  $d_1 - d_7$ , respectively to the inputs of the photodetectors F5-F11 of the central element and its neighboring elements to the left, right, above and below from the output of the light modulator M16 of the central element it is possible to compute the optical signal  $y_1 = \overline{x_1} \vee d_2$ , with  $M17 - y_2 = \overline{x_1} \vee x_2$ ,  $x_3 \vee x_4$ , with  $M18 - y_3 = \overline{x_4} \vee b_3$ , with  $M19 - y_4 = \overline{x_5} \vee x_6$ , with  $M20 - y_5 = \overline{x_6} \vee c_5$ , with  $M21 - y_6 = \overline{x_6} \vee a_7$ , with  $M22 - y_7 = \overline{x_6} \vee x_7$ .

We will consider the function of the matrix processor element for the example of the execution of the function

$$p_1 = \bigwedge_h [(a_h \sim b_h) \vee c_h] \quad p_2 = \bigwedge_h [(a_h \oplus b_h) \vee c_h]$$

where  $k = 1, 2, \dots, 5$  is the number of the element from the configuration including the central element and its four neighboring elements in the matrix. We will designate the central element  $i,j$ , the neighboring element to the left is  $i, j-1$ , to the right  $i, j+1$ , above  $i-1$ , below  $i+1, j$ .

The variable  $A\{a_{ij}\}$  is written in the first cell of the processor, the variable  $B\{b_{ij}\}$  in the third and the variable  $C\{c_{ij}\}$  enters in the form of optical signals into the inputs of the corresponding photodetectors of the converters during the data processing operation. The transform  $P\{p_{ij}\}$  is carried out simultaneously for all  $p_{ij}$ .

The algorithms for carrying out the functions  $p_1$  and  $p_2$ , which differ only in the first few steps, are given below.

For the function

$$p_1 = \bigwedge_h [(a_h \sim b_h) \vee c_h]:$$

Step 1. Switches K2, K4-K6 are closed on the power sources V2, V4-V6 and the capacitors C2, C4-C11 are charged.

Step 2. Switches K2, K4 are grounded. The optical signals  $a_{ij}$  and  $b_{ij}$  from the outputs of M12 and M14 to inputs of F2 and F4 are read.

Step 3. The signals  $\bar{a}_{ij}$  and  $\bar{b}_{ij}$  from the outputs of M12 and M13 to the inputs of F3 and F4 are read.

Step 4. Switch K5 is grounded. The signals  $\bar{a}_{ij} \vee b_{ij}$  and  $a_{ij} \vee \bar{b}_{ij}$  from the outputs of M14 and M15 to the inputs of F7 and F8 are read.

Step 5. Switch K5 is grounded. The signal  $(a_{ij} \sim b_{ij})$  from the output of M17 to the input of F10 is read. Signal  $c_{ij}$  goes to the input of F10.

For the function

$$p_2 = \bigwedge_h [(a_h \oplus b_h) \vee c_h]:$$

**Step 1.** Switches K2, K4-K6 close on the power sources V2, V4-V6.

**Step 2.** Switches K2, K4-K5 are grounded. The signals  $\bar{a}_{ij}$ ,  $\bar{b}_{ij}$  from the outputs of M12, M14 to the inputs of F2, F3 are read and the signal  $\bar{a}_{ij} \bar{b}_{ij}$  from the output of M14 to the input of F7 is read.

**Step 3.** The signal  $a_{ij} b_{ij}$  from the output of M17 to the input of F10 is read.

**Step 4.** Switch K6 is grounded. The signal  $(a_{ij} \oplus b_{ij})$  from the output of M17 to the input of M17 is read. The signal  $c_{ij}$  goes to the input of F10.

Thus, in five steps the function

$$p'_1 = |(a_{ij} \sim b_{ij}) \vee c_{ij}|$$

is formed and written in six cells of each element of the processor while four steps are required for the function

$$p'_2 = |(a_{ij} \oplus b_{ij}) \vee c_{ij}|$$

In the following steps, the algorithm for processing the configuration of k elements is the same for both functions. For example for function  $p_2$ :

**Step 5.** Switch K5 is closed on power source V5.

**Step 6.** Switch K5 is grounded. The signal  $\neg |(a_{ij} \oplus b_{ij}) \vee c_{ij}|$

from the outputs of M19-M22 to the inputs of F5-F8 respectively is read.

**Step 7.** Switch K6 is closed on the power source V6.

**Step 8.** Switch K6 is grounded. Reading is carried out of signals

$$\neg (\neg |(a_{ij} \oplus b_{ij}) \vee c_{ij}| \vee \neg |(a_{i,j+1} \oplus b_{i,j+1}) \vee c_{i,j+1}|)$$

from the output of M18 to the input of F1 and of

$$\neg (\neg |(a_{ij} \oplus b_{ij}) \vee c_{ij}| \vee \neg |(a_{i+1,j} \oplus b_{i+1,j}) \vee c_{i+1,j}|)$$

from the output of M16 to the input of F9.

**Step 9.** Switch K5 is closed on the power source V5.

**Step 10.** Switch K5 is grounded. The signals

$$\neg |(a_{i+1,j} \oplus b_{i+1,j}) \vee c_{i+1,j}| \vee \neg |(a_{ij} \oplus b_{ij}) \vee c_{ij}|,$$

$$\neg |(a_{i-1,j} \oplus b_{i-1,j}) \vee c_{i-1,j}| \vee \neg |(a_{ij} \oplus b_{ij}) \vee c_{ij}|,$$

$$\neg |(a_{i,j-1} \oplus b_{i,j-1}) \vee c_{i,j-1}| \vee \neg |(a_{ij} \oplus b_{ij}) \vee c_{ij}|,$$

$$\neg |(a_{i,j+1} \oplus b_{i,j+1}) \vee c_{i,j+1}| \vee \neg |(a_{ij} \oplus b_{ij}) \vee c_{ij}|$$

from the outputs of M19-M20 to the inputs of F5-F8 respectively are read.

**Step 11.** A reading is made of the result

$$p_2 = \neg (\neg |(a_{i+1,j} \oplus b_{i+1,j}) \vee c_{i+1,j}| \vee \neg |(a_{ij} \oplus b_{ij}) \vee c_{ij}| \vee \neg |(a_{i-1,j} \oplus b_{i-1,j}) \vee c_{i-1,j}| \vee \neg |(a_{i,j-1} \oplus b_{i,j-1}) \vee c_{i,j-1}| \vee \neg |(a_{i,j+1} \oplus b_{i,j+1}) \vee c_{i,j+1}|) = \bigwedge_h^b |(a_h \oplus b_h) \vee c_h|$$

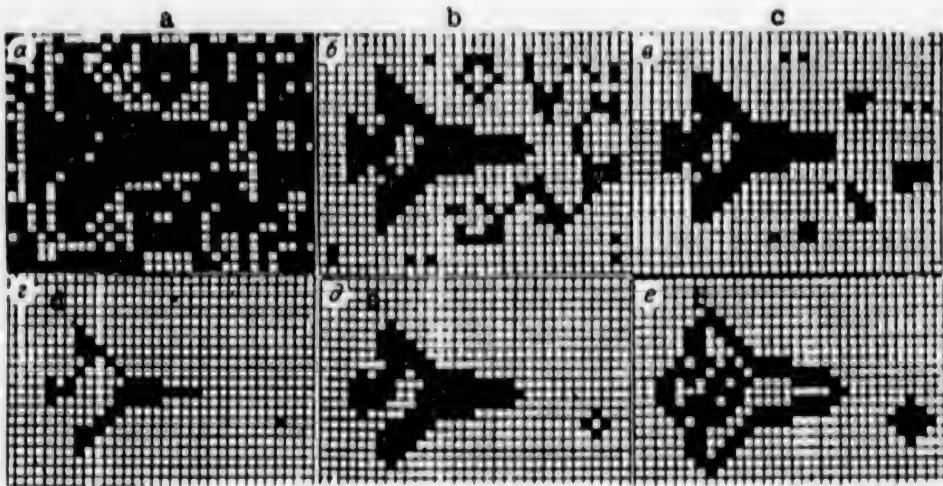


Figure 5

from the output of M17 to the input, for example, of F2 for further processing or for subsequent output by means of a shift to a peripheral unit.

By means of a sequence of transforms of the type  $P p_1$ , processing is carried out of binary images and the number and type of transforms are determined by the quality of the image and the assigned task and does not depend upon the dimensions of the image.

Fig. 5, a-f shows an example of image processing by a sequence of transforms of the indicated type: a is the initial image; b is noise filtering, the transform

$$p = \bigwedge_h^5 a_h,$$

execution time for the operation is 8 clock periods; c is noise filtering with masking, transform is

$$p = \bigvee_l^6 \left[ \bigwedge_h^5 (a_h \vee c_{hl}) \right],$$

execution time for operation is 48 clock periods; d is noise filtering, transform is

$$p = \bigwedge_h^5 a_h,$$

execution time for operation is 8 clock periods; e is broadening of lines,

$$p = \bigvee_l^4 \left[ \bigwedge_h^5 (a_h \vee c_{hl}) \right],$$

time for executing operation is 32 clock periods; f is discrimination of outline,

$$p = \neg \left[ \left( \bigwedge_h^5 a_h \right) \vee \left( \bigwedge_h^5 \bar{a}_h \right) \right],$$

time for executing operation is 18 clock periods.

In addition to the realization of the indicated functions, the processor with dimensionality  $[m \times m]$  designed on the basis of the described element is capable of storing 5 data arrays  $[m \times m]$ , simultaneously storing and transforming three arrays  $[m \times m]$ , shifting data arrays  $[k \times k]$ ,  $1 \leq k \leq m$  in the matrix in four directions, carrying out a complete system of logic functions of two variables, operating pairwise addition of  $2m$  numbers with word lengths  $m - 1$  and a series of other functions.

Analysis of the special features of the functioning of the described unit, its design characteristics and comparison with modern microelectronic units capable of carrying out analogous functions make it possible to draw the following conclusions:

1. The functional element of the optical processor contains a significantly smaller number of component elements (two orders less according to evaluations).
2. The overall number of clock periods expended for execution of the required logic operations, in the given case, for image processing, is also smaller, at least  $N$  times less ( $N$  is the number of lines in the imager) and the period for complete processing of the images does not depend its dimensions.

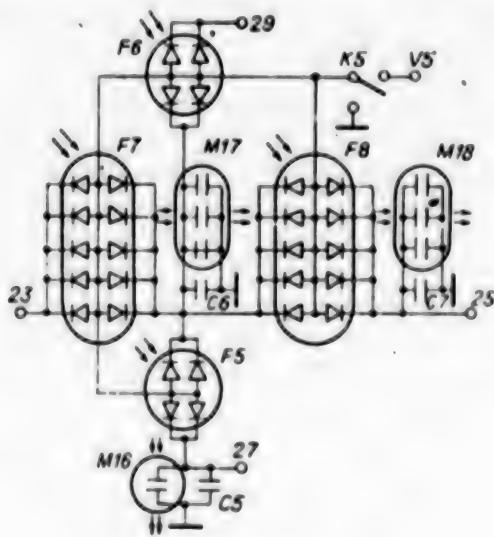


Figure 6

3. The area occupied by the connections (see the block diagram (Fig. 6) and the topological fragment (Fig. 7) of the cell of the functional converter) comprises, as shown by analysis, 7-8% of the overall area of the substrate (in modern microcircuits 90% [5]).

In addition, the described units are distinguished by the greater volume density of the element arrangement, the reliability of the connections, noise stability and, finally, by suitability for industrial production.

The indicated advantages are the result of the different nature of the energy transmission on the supply buses and signal channels, the absence of rectifier and switch elements in the logic link channels (due to the specific features of the propagation of light flux), the combination of memory and logic functions and more effective execution of computing procedures which is attained through the three-dimensional design of the element in which the entire array of information is processed simultaneously without sorting into separate parts and preliminary storage of the corresponding groups of numbers with nonproductive expenditure of time. As a quantitative illustration of the effectiveness of increasing the number of optical channels we will compare the described element with an element of a cellular unit

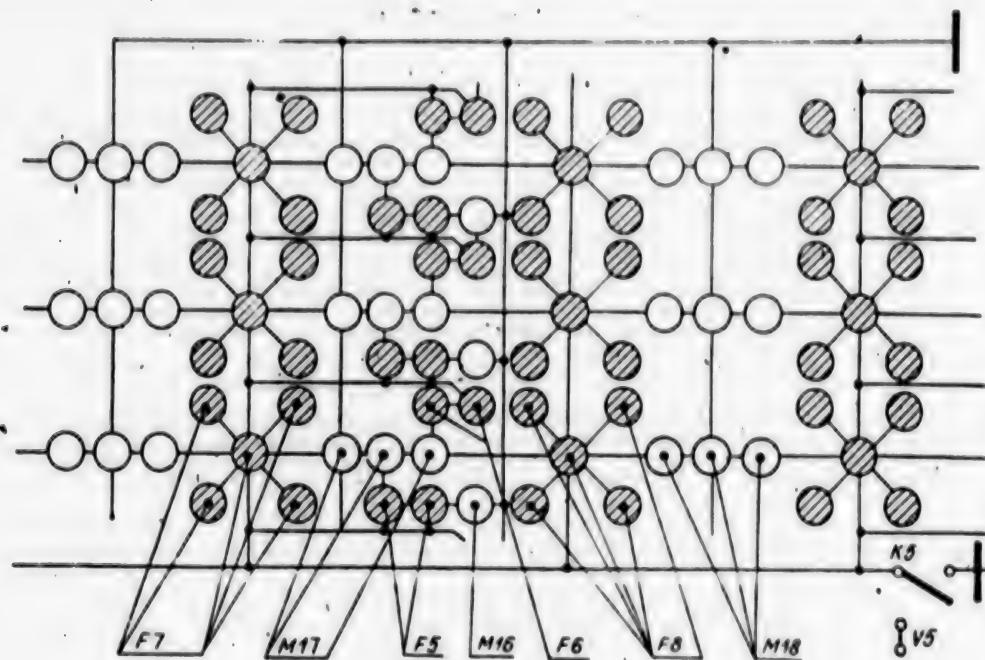


Figure 7

using the same element base [19]. The latter is also distinguished by the small number of components, i.e., 200, has 31 optical channel links and carries out only one function  $P_1$ , and the memory function. The element described above contains 58 optical channel links and consists of only 119 components but has a significantly wider range of functional possibilities.

What are the possibilities for the practical realization of the described element base? Reports have been made on experimental research on a mock-up of a dynamic memory designed on the basis of electrooptic crystals of lithium niobate [21]. A quantitative relation was established for the experimentally obtained and theoretical transfer characteristics for a voltage  $V = 0.3V_0$  ( $V_0$  is the half-wave voltage) into the modulator. This value is greater than the voltage values which are characteristic for modern microelectronic elements. Therefore research was carried out in order to study the possibilities for further lowering of the value of  $V$ , and it was experimentally established that when a photoelectric converter with switch characteristics is used the functional capacity of the memory cell is preserved at  $V = 0.03V_0 \lambda/2$  (in the real experiment 5V).

The fabrication of the computers described above with element arrangement density compatible with the density of elements in VLSI requires the creation of a thin-film light modulator which is a quite complex fabrication task. The first publication of a report on the observation of a longitudinal electrooptical effect in thin ferroelectric strontium barium niobate films is [22]. It was later possible to obtain a value for the electrooptic coefficient  $r_{ef}$  compatible with the value of  $r$  in strontium barium niobate and to use thin-film structures (transparent electrode - ferroelectric film - transparent electrode) for pulse modulation of light flux with modulation depth of 80-90% and rotation of the plane of polarization  $\phi$  of the light of  $2-3^\circ$  [23]. This value for  $\phi$  is already sufficient for practical use (see, for example, [24] where magneto-optical memory units of a commercial scale are described in which the value  $\phi \approx 0.3 - 0.6^\circ$ ). In addition, as analysis shows, there is a reserve for an increase in the value of  $\phi$  by at least one order of magnitude by means of a variation in the composition of the ferroelectric and the introduction into the film of rare earth element oxides, for example, lanthanum oxide [25].

It is thus possible to assert that contemporary microelectronic technology supplies the fundamental possibility for fabricating thin-film electrooptic light modulators with control voltages which are close to those which are characteristic for microelectronic elements.

It follows from the above presentation that a transition to the new element base will not require large capital investments.

#### BIBLIOGRAPHY

1. Solomon P.M., "Sравнение полупроводниковых приборов для скоростных логических схем" [Comparison of Semiconductor Devices for High-Speed Logical Circuits], TIIER, Vol 70, No 5, pp 88 - 112, 1982.
2. Fraser D.L. et al. "Gigabit Logic Circuits with Scaled NMOS", in ESSARC, Dig. Techn. Paper, 1981, pp 202-204.
3. Abe M., T. Mimura, N. Yokoyama and H. Isikawa, "New Technology toward GaAs LSI/VLSI for Computer Applications", Trans. Electron. Dev. and IEEE Trans. Microwave Theory Techn., July 1982.
4. "Novaya moshchnaya model' serii superkomp'yuterov firmy "CRAY"" [New Powerful Model of Supercomputer Series of "CRAY" Company], ELEKTRONIKA, No 9, p 6, 1982.
5. Dao T.T., IEEE COMPCON, Spring, p 184, 1981.
6. Kiiz R.U., "Fundamental'nyye predely v tsifrovoy obrabotke informatsii" [Fundamental Limits to Digital Data Processing], TIIR, Vol 69, pp 152-166, 1981.
7. Sumhey L.W., IEEE SPECTRUM, Vol 17, No 4, P 24, 1980.
8. Chiba T., "Impact of the LSI on High-Speed Computer Packaging", IEEE Trans. Comput., Vol C-27, pp 319-325, 1979.
9. Landman B.S. and R.L. Russo, "On a Pin versus Block Relationship for Partitions of Logic Graphs", IEEE Trans. Comput., Vol C-20, pp 1469-1479, 1971.
10. Russo R.L., "On the Trade-Off between Logic Performance and Circuit-to-Pin Ratio for LSI", IEEE Trans. Comput., Vol C-21, pp 147-153, 1972.
11. Yelinson M.I. and A.A. Sukhanov, "Problemy mezhsoyedineniy v sovremennoy mikroelektronike" [Interconnection Problems in Modern Microelectronics], MIKROELEKTRONIKA, Vol 13, No 3, pp 179-195, 1984.
12. Lou A. "Fizicheskaya realizatsiya tsifrovyykh logicheskikh skhem" [Physical Realization of Digital Logic Circuits], Moscow, Sovetskoye Radio, 1967, pp 30-55.
13. Andrushko L.M., V.A. Voznesenskiy and I.P. Panfilov, "Sovremennoye sostoyaniye i perspektivy razvitiya opticheskikh integral'nykh skhem" [Contemporary Status and Prospects for the Development of Optical Integrated Circuits], ZARUBEZH. RADIOELEKTRONIKA, No 11, pp 60-71, 1983.

14. Sveshnikov S.V., "Elementy optoelektroniki" [Elements of Optoelectronics], Moscow, Sovetskoye Radio, 1977.
15. Kosnocky W.F., "Laser Digital Devices", in Internat. Solid-State Circuit Conf., N.Y., 1965, p 269.
16. Eybrekhem A., K.T. Siton and S.D. Smit, "Opticheskiy komp'yuter" [Optical Computer], V MIRE NAUKI, No 4, pp 15-25, 1983.
17. "Opticheskiy tranzistor, rabotayushchiy pri komnatnoy temperaturе" [Optical Transistor Operating at Room Temperature], ELEKTRONIKA, No 6, p 17, 1983.
18. "Vazhnyye komponenty opticheskikh skhem" [Important Components of Optical Circuits], ELEKTRONIKA, No 26, p 3, 1982.
19. Kostsov E.G. and A.I. Mishin, "Osobennosti postroyeniya opticheskikh TsVM" [Special Features of the Design of Optical Computers], MIKROELEKTRONIKA, Vol 6, No 2, pp 139-151, 1977.
20. Kostsov E.G., V.K. Malinovskiy, Yu.Ye. Nesterikhin and A.N. Potanov, "Osobennosti fizicheskoy realizatsii operativnoy opticheskoy pamyati" [Special Features of the Physical Realization of an Operational Optical Memory], AVTOMETRIYA, No 4, pp 3-7, 1976.
21. Kostsov E.G. and A.N. Potanov, "Porogovyy logicheskiy element" [Threshold Logical Element], AVTOMETRIYA, No 5, p 93, 1976.
22. Baginsky I.L. et al, "Some Peculiarities of Strontium Barium Niobate Films and their Electrophysical Properties", FERROELECTRICS, Vol 22, p 783, 1978.
23. Antsygin V.D., E.G. Kostsov and L.N. Sterelyukhina, "Impul'snaya elektro-opticheskaya modulatsiya sveta v tonkikh segnetoelektricheskikh plenkakh" [Pulse Electric Modulation of Light in Thin Ferroelectric Films], AVTOMETRIYA, No 5, p 98, 1983.
24. "Chetyrekhloynyy magnitoopticheskiy disk" [Four-Layer Magnetooptic Disk], ELEKTRONIKA, Vol 55, No 14, p 16; Vol 55, No 15, p 11, 1983.
25. Lui S.T. and A.S. Bhalla, "Some Interesting Properties of Dislocation-Free and La-Modified  $\text{Sr}_{0.5}\text{Ba}_{0.5}\text{Nb}_2\text{O}_6$ ", FERROELECTRICS, Vol 51, pp 47-51, 1983.

COPYRIGHT: Izdatel'stvo "Nauka", "Avtometriya", 1985

12497  
CSO: 1863/266

'START' CONSORTIUM PUSHES MARS, FIFTH-GENERATION DEVELOPMENT

Tallin SOVETSKAYA ESTONIYA in Russian 14 Jun 85 p 1

[Text] Scientists of the Estonian Academy of Sciences' Institute of Cybernetics have devoted much attention to documents of the conference on questions of accelerating scientific-technical progress which took place in the Central Committee of the Communist Party of the Soviet Union. Comrade M. S. Gorbachev's speech pointed out that microelectronics, computer technology, instrument building and the whole information-science industry are a catalyst of progress. They require accelerated development. This requirement poses new tasks for cyberneticists, and these tasks were discussed by E. Tyugu, academician-secretary of the Estonian academy's recently created department of information science and technical physics:

"Our department was created for the purpose of improving economic-organizational forms of integration of research in a key direction of science. Our current objective is to develop computers with new architecture and with artificial-intelligence software and hardware, as well as appropriate system and applied software. Basic knowledge and practical experience have already been amassed in this work, and they must now be utilized. It is for this purpose that a temporary scientific-technical group called 'Start' has been organized by a decision of the USSR State Committee for Science and Technology and the presidium of the USSR Academy of Sciences. This group takes in the computer center of the USSR Academy of Sciences' Siberian Branch, the academy's Moscow Computer Center, the Estonian academy's Institute of Cybernetics, and the Severodonetsk Scientific Research Institute of Control Computers, which belongs to the 'Impul's' Research and Production Association of the USSR Ministry of Instrument Building, Means of Automation and Control Systems. Considerable forces have been assembled, and they are called upon to shorten the time of the 'research--development--introduction' cycle and to expedite the development of components for a new system called 'Mars' which will be up to the standard of the best world models. I am confident that the tasks which the Party has assigned to scientists for the acceleration of scientific-technical progress will be accomplished. Joining the work of 'Start', we have pledged to develop a mockup of the 'Mars' system by 1988."

FTD/SNAP

'DISK' COMPLEX HELPS INTERPRET WEATHER SATELLITE DATA

Moscow IZVESTIYA in Russian 10 Jun 85 p 1

[Article by correspondent V. Reznik]

[Text] Khabarovsk--Specialists from the German Democratic Republic have installed a hardware complex called "Disk."

"The 'Disk' is a joint development of partner countries of the Council for Mutual Economic Assistance," related Helmut Wolbing, the project's technical director. "The concept of the complex, its software and its principal microcircuits belong to our Soviet friends. A number of its components were made in the People's Republic of Bulgaria."

The "Disk" will help in rapid interpretation of satellite information, on the basis of which accurate forecasts of weather and various natural phenomena on the territory of the Far East can be made.

CSO: 1863/353  
FTD/SNAP

UDC 681.324

## ELEKTRONIKA MS 1603 HIGH-SPEED PERIPHERAL PROCESSOR

Novosibirsk AVTOMETRIYA in Russian No 2, Mar-Apr 85 (manuscript received 25 Oct 84) pp 88-90

[Article by V. A. Dyboy, V. V. Kashtanov, V. O. Lazarev and A. A. Fokin (Voronezh) under the rubric: "Brief Reports"]

[Text] High-speed peripheral processors, with their high computation speed, high reliability, compact design and low power requirement, have found wide application in recent years. An example of such a system is the Elektronika MT-70 high-speed peripheral processor, series-produced by domestic industry for many years.

The Elektronika MS 1603, developed on the basis of operational experience with the MT-70 and compatible with the latter at the application program level, is a representative of the new generation of high-speed peripheral processors.

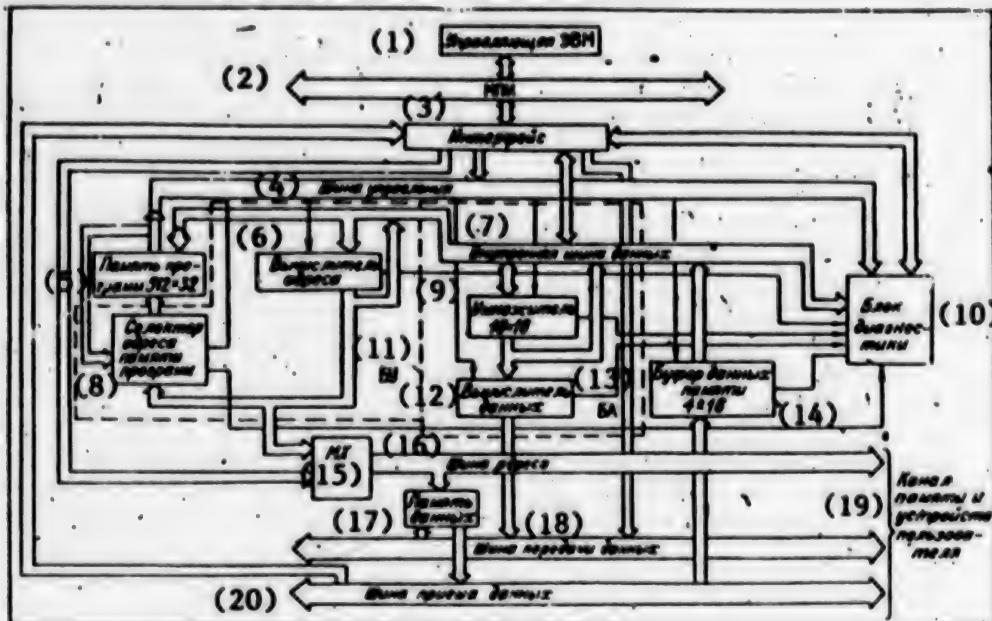
The Elektronika MS 1603 high-speed peripheral processor can be used in conjunction with any computer having a common-bus type parallel bus interface between modules. Its primary purpose in operation with an Elektronika 60 computer system is the execution of a large number of vector multiplication and addition operations.

The technical specifications of the Elektronika MS 1603 high-speed peripheral processor are as follows: 16-bit word length; fixed point numeric representation; main memory program capacity of 512 32-bit words; main memory data capacity of 32K-words; memory cycle time of 200 ns; and parallel multiplication and addition operation execution time of 200 ns.

The Elektronika MS 1603 high-speed peripheral processor has a clock frequency twice that of the Elektronik MT-70 and a size and power consumption level 1.5-fold as great. The Elektronika MS 1603 high-speed peripheral processor's performance is twice that of its predecessor in executing typical tasks (in particular, its fast Fourier transform execution time on an array consisting

\* B. L. Tolstykh, I. G. Talov, V. V. Plotnikov and G. V. Bondarovich.  
"Elektronika MT-70 High-Speed Peripheral Processor," USiM, 1983, No 4,  
p 122-125.

of 1024 complex numbers is less than 11 ms). The improvement is due to the use of new circuit engineering solutions and modern components, specifically the KM-1804 microprocessor unit and KM 1802 VR5 16-bit multiplier large-scale integrated circuit. The illustration shows the block diagram of the Elektronika MS 1603 high-speed peripheral processor.



Elektronika MS 1603 high-speed peripheral processor block diagram

- |                                      |   |
|--------------------------------------|---|
| 1—Master computer                    | 11—Control unit                         |
| 2—Parallel bus interface             | 12—Data calculation unit                |
| 3—Interface                          | 13—Arithmetic unit                      |
| 4—Control bus                        | 14—Memory data buffer ( $4 \times 16$ ) |
| 5—Program memory ( $512 \times 32$ ) | 15—Multiplexer                          |
| 6—Address calculation unit           | 16—Address bus                          |
| 7—Internal data bus                  | 17—Data memory                          |
| 8—Program memory address selector    | 18—Data transmission bus                |
| 9—Multiplier ( $16 \times 16$ )      | 19—Memory and user device channel       |
| 10—Diagnostic unit                   | 20—Data reception bus                   |

All high-speed peripheral processor busses have a capacity of 16 bits and are fitted with tri-state devices at their outputs. The internal data bus is used to transfer data between all high-speed peripheral processor units except data memory. The control bus is used to move instruction codes from program memory or the interface read-only memory [ROM] to the control unit and arithmetic unit. The memory and user device channel is a system of busses assuring data/memory communications with the parallel interface bus, via the interface unit and with the high-speed peripheral processor's arithmetic unit. The memory and user device channel supports an information exchange rate of 10Mbytes/second. Additionally, the memory and user device channel can be used to connect user devices, such as digital-to-analog converters, analog-to-digital converters, external memory, etc., to the high-speed peripheral processor. These devices can be accessed by the processor as easily as data

memory. An interrupt and priority handling unit on the channel is used to resolve conflicts which can arise when multiple user devices are connected to the high-speed peripheral processor.

Data memory consists of a main memory unit with a minimum capacity of 32K-words. The control computer can use data memory in a direct-access mode or in a program mode. The arithmetic unit can also access data memory in the same manner, with control computer/data memory exchanges proceeding in parallel with high-speed peripheral processor program execution.

The arithmetic unit is designed to carry out various arithmetic and logic operations on data. Processing operands can arrive at the unit on the internal bus from data memory via the buffer used for preselection and from other high-speed peripheral processor units. Operation results are transmitted on the internal data bus or on the memory and user device channel.

The control unit includes an address calculation unit and program address selector. The first unit is primarily used to compute data and program memory addresses although it can also be used as a supplementary calculator to organize parallel computation within the confines of a single instruction.

Program memory consists of a ROM unit with a capacity of 512 32-bit words designed to store the codes of instructions to be executed in the high-speed peripheral processor. A high-speed peripheral processor's instruction word consists of 32 bits and is transmitted via the 16-bit control bus in two parts during a single processor clock cycle. New data can be loaded in program memory from the master computer, from data memory or from any high-speed peripheral processor device via the internal data bus. Program memory data can be used as operands when they are transmitted via the internal data bus.

The interface links the high-speed peripheral processor to the master computer and contains the hardware and software resources needed to control the high-speed peripheral processor's operation as well as data exchanges between the parallel bus interface and the high-speed processor. High-speed peripheral processor operational and data exchange control is accomplished by means of these directly addressable registers included in the interface (register octal addresses are given in parentheses): word counter register for the direct memory access mode (176100); channel address register for the direct memory access mode (176102); direct memory access control register (176104); data memory address register (176106); register for data exchanges with the high-speed peripheral processor's central processing unit (176110); panel operation register (176112); register for data exchanges between the control computer and data memory (176114); and the diagnostics register to control high-speed peripheral processor's built-in diagnostics functions (176116).

In addition to the system registers, the interface contains a panel operation ROM. Panel operation codes are sent from the ROM via the control bus in the same manner as instruction codes from program memory. Panel operations allow the computer operator to exchange data with high-speed peripheral processor components which do not have system-wide addresses.

Item number	Algorithm designation	Processing time for an array consisting of 1023 complex points (ms)
1	Addition of arrays	0.9
2	Addition of array and constant	0.7
3	Subtraction of arrays	0.9
4	Subtraction of constant from array	0.7
5	Multiplication of arrays	0.9
6	Multiplication of array by constant	0.7
7	Double-precision multiplication of arrays	1.1
8	Double-precision multiplication of array by constant	0.9
9	Logical ANDing of arrays	0.9
10	Logical ANDing of array with constant	0.7
11	Logical ORing of arrays	1.1
12	Logical ORing of array with constant	1.1
13	Exclusive ORing of arrays	0.7
14	Exclusive ORing of array with constant	0.7
15	Complex multiplication of arrays	1.3
16	Multiplication of array by complex/conjugated arrays	1.3
17	Right shift of array	0.7
18	Left shift of array	0.7
19	Determination of array maximum	1.1
20	Determination of array minimum	1.1
21	Convolution	14.7
22	Correlation	14.7
23	Array transmission	0.7
24	Fast Fourier transform (FFT)	15
25	Reverse fast Fourier transform (RFFT)	15
26	Packing for FFT	0.9
27	Packing for RFFT	0.9
28	Unpacking for FFT	1.4
29	Unpacking for RFFT	1.4
30	Clearing	0.7
31	Computation of spectral amplitudes	0.9
32	Dual inversion of array	2.0

The diagnostics unit provides access for test programs to the high-speed processor's internal busses and test points.

The high-speed peripheral processor's instruction set actually consists of the set of operations carried out by the individual units. This provides a flexible system which can be modified depending on the requirements of specific tasks. This, combined with the presence of a main memory unit for programs, provides the user with a wide variety of capabilities for the development and debugging of original programs. The listing above shows the standard algorithms available to the user in the Elektronika MS 1603 high-speed peripheral processor's instruction set.

The Elektronika MS 1603 high-speed peripheral processor's high performance characteristics, compact design and ease of servicing assure its widespread application in many fields of science and technology where high reaction speed is required together with a moderate level of precision in computation.

COPYRIGHT: Izdatel'stvo "Nauka", "Avtometriya", 1985

12746

CSO: 1863/355

UDC 621.396.6

## THE 15UT-4-017 COMPLEX AS A WORKSTATION FOR CIRCUIT ENGINEERING MODELING

Novosibirsk AVTOMETRIYA in Russian No 2, Mar-Apr 85 (manuscript received 5 Oct 83, final version 29 Oct 84) pp 95-97

[Article by V. Ye. Mezhov (Voronezh) under the rubric: "Brief Reports"]

[Text] The Elektronika 100-25 minicomputer-based general-purpose 15UT-4-017 unit [1], has found widespread application as a standalone or integrated system for designing electronic products. In practice, the system is used to automate the routine work of entering, checking, editing, storing, retrieving, converting and documenting alphanumeric and graphic information and to solve rather complex engineering problems such as the design of integrated-circuit topology, microwave transistors, printed circuit boards and circuit engineering [2,3]. In particular, work [3] examines the system's application software designed to calculate digital circuit electrical characteristics. Experience shows, however, that relatively high computation times are required for the calculation of typical cell and functional block electrical characteristics. This precludes the possibility of implementing design work on an interactive basis, i.e. at a rate commensurate with human designer reactions.

The development of the Elektronika MT-70M high-speed peripheral processor, described in [4], provided a solution to this problem. This processor has a parallel structure and a sophisticated bus-type organization of data and instruction communications between individual units and can carry out pipelined processing of data arrays. The inclusion of the Elektronika MT-70M processor in the 15UT-4-017 unit (as an external device for the Elektronika 100-25 computer), significantly expanded the system's computational resources.

Some characteristics of the use of a peripheral processor in electrical property analysis problems are discussed in the work cited.

The article [3] describes the PRATsIS application program package implemented on the 15UT-4-017 unit. When this package is used to calculate the dynamic characteristics of circuits, a mathematical model of the circuit under analysis is presented in the form of a system of ordinary differential equations, solved by means of an implicit second-order integration method. In this method, quantization of the initial ordinary differential equation system at each time step leads to a system of non-linear algebraic equations. Their

linearization (with Newton's method computation of corrections) results in the solution of a linear equation system upon each integration step. This requires the formation of unit admittance arrays and a unit current vector (through sequential referencing of circuit component models).

Table 1. Computation times for transition characteristic calculation

Circuit complexity (Y=number of units, K=number of components)	15UT-4-017		BESM-6	
	Total <u>time</u>	Linear equation system formation and solution <u>time</u>	Total <u>time</u>	Linear equation system formation and solution <u>time</u>
Y = 7 K = 8	115 sec.	102 sec.	35 sec.	30 sec.
Y = 52 K = 76	231 min.	204 min.	50 min.	43 min.
Y = 55 K = 89	174 min.	136 min.	42 min.	35 min.

The analysis of computation times for the solution of transition processes on the basis of implicit integration methods shows (Table 1) that approx. 90 percent of the time is taken up in forming and solving linear equation systems. The use of a high-speed peripheral processor to execute these repeated and truly vectored computation steps allowed a sharp reduction in analysis time. Table 2 gives the results of experimentation using the modified PRATsIS-M program package (for a 15UT-4-017 unit with an Elektronika MT-70M processor) to design integrated circuits of varying degrees of complexity together with the same parameters for programs operating on the BESM-6 computer. These data show that inclusion of a peripheral processor in the system allowed a capacity increase greater than one order of magnitude and provided better time figures in comparison to the high-performance BESM-6 computer.

Now the PRATsIS-2 logic modeling program package has been developed for the 15UT-4-017 unit. This package allows the following functions: input and processing of descriptions of circuits and their output activities; logic modeling of circuit operations over time; and comparison between expected and steady-state circuit reaction to the input activity vector. The component model library contains 24 typical units (AND-, NAND- and OR-gates, various types of one-shots, etc.) as well as a number of small-, medium and large-scale integrated microcircuits, allowing the modeling of a wide variety of circuits varying in their purpose and complexity. The program includes an interpretive-type event algorithm and a four-place modeling alphabet (0, 1 are the logic "zero" and "one" values, X is an undetermined state and Z is a high-impedance state). Also provided is a means of producing microcircuit operation maps for the digital component developed.

Table 2. Machine time involved in calculating transition processes

Circuit complexity (Y=number of units, X=number of components)	Number of generators	Transition process duration (ms)	<u>Calculation time</u>		
			<u>15UT-4-017</u>	<u>+Elektronika MT-70M</u>	<u>BESM-6</u>
Y = 10 K = 16	1	50	33 sec.	3 sec.	4 sec.
Y = 14 K = 20	3	16,000	13 min. 46 sec.	1 min. 7 sec.	2 min.
Y = 22 K = 36	3	16,000	45 min.	2 min. 59 sec.	6 min.

Table 3. Computational efforts involved in logic modeling of digital circuits

System characteristics	15UT-4-017 (Elektronika 100-25)			<u>Elektronika-79</u>		
	A	B	C	A	B	C
107 typical elements						
563 lines						
6 inputs	314.2	78	203	229.42	108	279
1200 -- modeling time (cycles)						
63625 -- number of library references						
27706 -- number of events						
800 typical elements						
806 lines						
6 inputs	728.8	57	252	334.39	125	334
27000 -- modeling time (cycles)						
183899 -- number of library references						
41999 -- number of events						
392 typical elements						
752 lines						
7 inputs	81.4	34	130	59.83	46	179
10000 -- modeling time (cycles)						
10552 -- number of library references						
2787 -- number of events						

Legend for Table 2:

- A--Modeling time, sec.
- B--Modeling rate, events/sec.
- C--Reference rate, models/sec.

PRATsIS-2 operates under the MDOS-RV [multiprogramming real-time disk operating system] and has been adapted for use on various Elektronika-family computers.

PRATsIS-2 can model the logic of circuits with up to 2000 components or 400 microcircuits (when running on a computer with an on-line memory capacity of 32K-words).

Table 3 shows the characteristics of PRATsIS-2 package in modeling circuits with different degrees of complexity. The data show that the 15UT-4-017 unit can be used successfully to perform logic modeling tasks.

#### BIBLIOGRAPHY

1. Mezhov, V. Ye., Ratmirov, N. L., Talov, I. L. and Tolstykh, B. L. "Use of an Elektronika 100-25 Minicomputer in an Automated System for Designing Circuit Topology," ELEKTRON. PROM-ST', 1978, No 10 (70).
2. Mezhov, V. Ye., Ratmirov, N. L., Talov, I. D. and Tolstykh, B. L. "15UT-4-017 System Software," ELEKTRON. PROM-ST', 1979, No 6 (78).
3. Mezhov, V. Ye., Talov, I. L. and Chernyayev, Yu. N. "15UT-4-017 System Circuit Engineering Design Application Software," ELEKTRON. PROM-ST', 1980, No 7 (91).
4. Tolstykh, B. L., Talov, I. L., Plotnikov, V. V. and Bondarovich, G. G. "Elektronika MT-70 High-Speed Processor," USiM, 1983, No 4.

COPYRIGHT: Izdatel'stvo "Nauka", "Avtometriya", 1985

12746

CSO: 1863/355

UDC 681.325.5

## DIAGNOSTICS OF PERIPHERAL PROCESSORS AND CONTROLLERS

Novosibirsk AVTOMETRIYA in Russian No 2, Mar-Apr 85 (manuscript received 25 Oct 84) pp 90-93

[Article by V. A. Dyboy, O. S. Semenova and A. A. Fokin (Voronezh) under the rubric: "Brief Reports"]

[Text] One of the problems facing computer system manufacturers and users is that of calibration and maintenance. The problem is caused by the increasing complexity of equipment combined with the inflexible nature of the methods of testing employed. These methods are based on component troubleshooting (the location and correction of failed components) according to indirect signs generated as the result of operational monitoring.

Under these conditions, the introduction of more progressive and higher quality (and more complex) equipment is inefficient due to the concomitant need for hiring or training highly qualified specialists to calibrate (during production) or repair (during operation) this equipment.

Using the example of the Elektronika MS 1603 high-speed peripheral processor,\* this article describes a diagnostic method using built-in diagnostic hardware which allows a significant increase in the scope of troubleshooting.

A general picture of the Elektronika MS 1603 high-speed peripheral processor's structure is required for a proper understanding of the diagnostic method described (see Fig. 1).

As the block diagram shows, the high-speed processor was designed around the block bus principle, i.e. it consists of four functionally independent units interconnected by data and control busses. Each functional unit is implemented in a separate block of components.

In addition to the busses, several separate lines carrying miscellaneous control and status data also interconnect the high-speed peripheral processor's units (indicated by the thin lines on the diagram). The high-speed

---

\* V. A. Dyboy, V. V. Kashtanov, V. O. Lazarev and A. A. Fokin. "Elektronika MS 1603 High-Speed Peripheral Processor," AVTOMETRIYA, 1985, No 2.

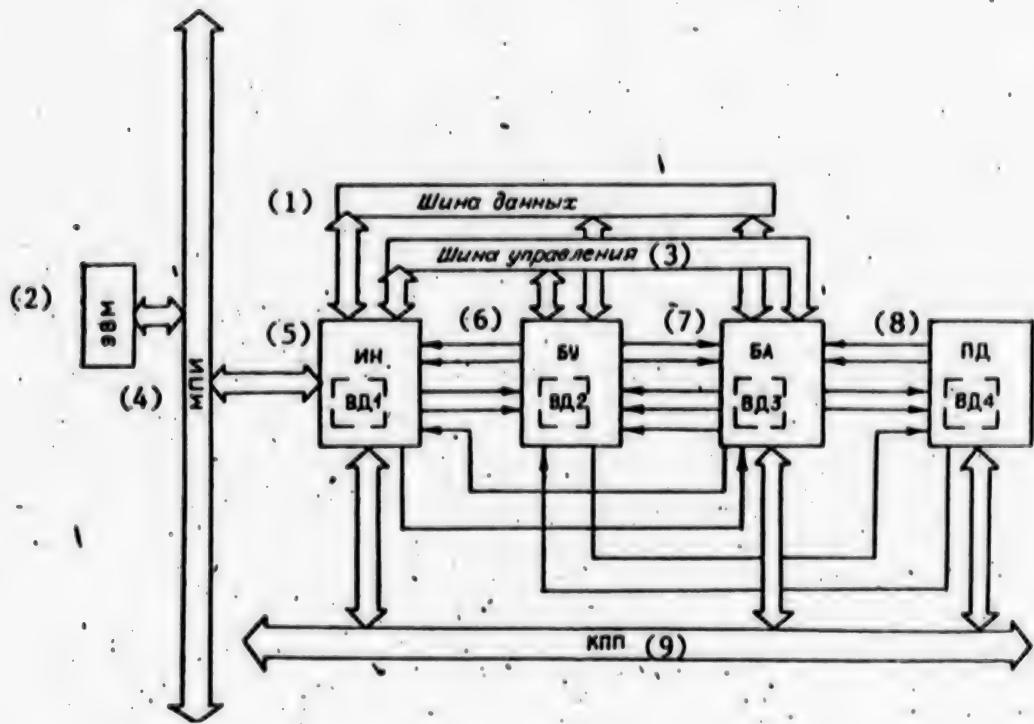


Figure 1. Elektronika MS 1603 high-speed peripheral processor block diagram

- |                            |                                  |
|----------------------------|----------------------------------|
| 1--Data bus                | 7—Arithmetic unit                |
| 2—Computer                 | Built-in diagnostic unit 3       |
| 3—Control bus              | 8—Data memory                    |
| 4—Parallel bus interface   | Built-in diagnostic unit 4       |
| 5—Interface                | 9—Memory and user device channel |
| Built-in diagnostic unit 1 |                                  |
| 6—Control unit             |                                  |
| Built-in diagnostic unit 2 |                                  |

peripheral processor is a completely synchronous device, except for that part of the interface which provides the link to the control computer. The high-speed peripheral processor's primary clock signal has a duration of 200 ns. The control computer manages high-speed peripheral processor operations and also handles its diagnostics.

The demands imposed on the high-speed peripheral processor's diagnostic system during its design can be summarized under the following three points:

1. The component block as the minimum level of troubleshooting.
2. The component in the component block as the maximum extent of troubleshooting.
3. Troubleshooting and fault isolation should take place without manual application of test programs.

To provide even a minimum level of troubleshooting, the search for a fault location must be made on the basis of direct rather than indirect indicators (as normally used). Two related tasks occur in this connection:

1. Provision must be made for access by diagnostic programs to all, or (at least) the primary, links between possible sources of faults at the desired troubleshooting level.
2. The diagnostic system must be provided with the capability of checking or storing test point states at the time an error occurs, i.e. at practically any moment during the test sequence.

This can be achieved only when built-in diagnostic tools are present in the high-speed peripheral processor itself. With these tools the control computer can access: the control bus used to transmit instruction codes from program memory to all high-speed peripheral processor elements; the internal data bus which can be used to send data from any high-speed peripheral processor element; and the nearly 50 separate control and status lines interconnecting the various blocks of components.

The transmission of data to the control computer from the high-speed peripheral processor's busses does not require a significant amount of hardware. The data bus and control bus are multiplexed in the interface unit and information is sent to the computer via one of the high-speed peripheral processor's system registers, which in the diagnostic mode take on a function different from that in the basic operating mode.

A shift register is fitted to each component block in order to transmit control line status information to the computer. All the shift registers are connected in series (see Fig. 2). In this manner, the status of all lines is sent to the computer sequentially via one of the diagnostic register's bits. The diagnostic register will be examined later. Data storage and transmission via the shift register take place in the following manner: at a given moment data from all the lines is simultaneously latched in the register and each sequential read of the diagnostic register causes a one-bit shift of data in the shift register.

High-speed processor operations are synchronized with those of the relatively slow computer by means of the high-speed peripheral processor's programmable clock signal generator [clock]. In the normal mode, the clock, activated by the high-speed peripheral processor's power supply voltage, is not interrupted. In the diagnostic mode, the clock can be stopped if the high-speed peripheral processor's state is fixed and cannot be changed as the result of stopping the clock (a "Halt" state for example). After stopping in the diagnostic mode, the clock can be programmed to execute a given number of cycles at its normal frequency. The last clock signal pulse drop prior to halting the clock carries information to the shift register and data from the data bus to the register for transmission to the computer. Thus, "steps" storing dynamic elements can be used to execute any high-speed peripheral processor function or the entire test sequence analyzing the status of the processor's internal busses and control lines during the intervals between "steps".

The Elektronika MS 1603 high-speed peripheral processor's clock can be programmed to work within the range between a half-cycle (the clock signal's shift from one logic level to the other) and seven cycles. Any operation in the high-speed peripheral processor is executed in a time ranging from one to three cycles.

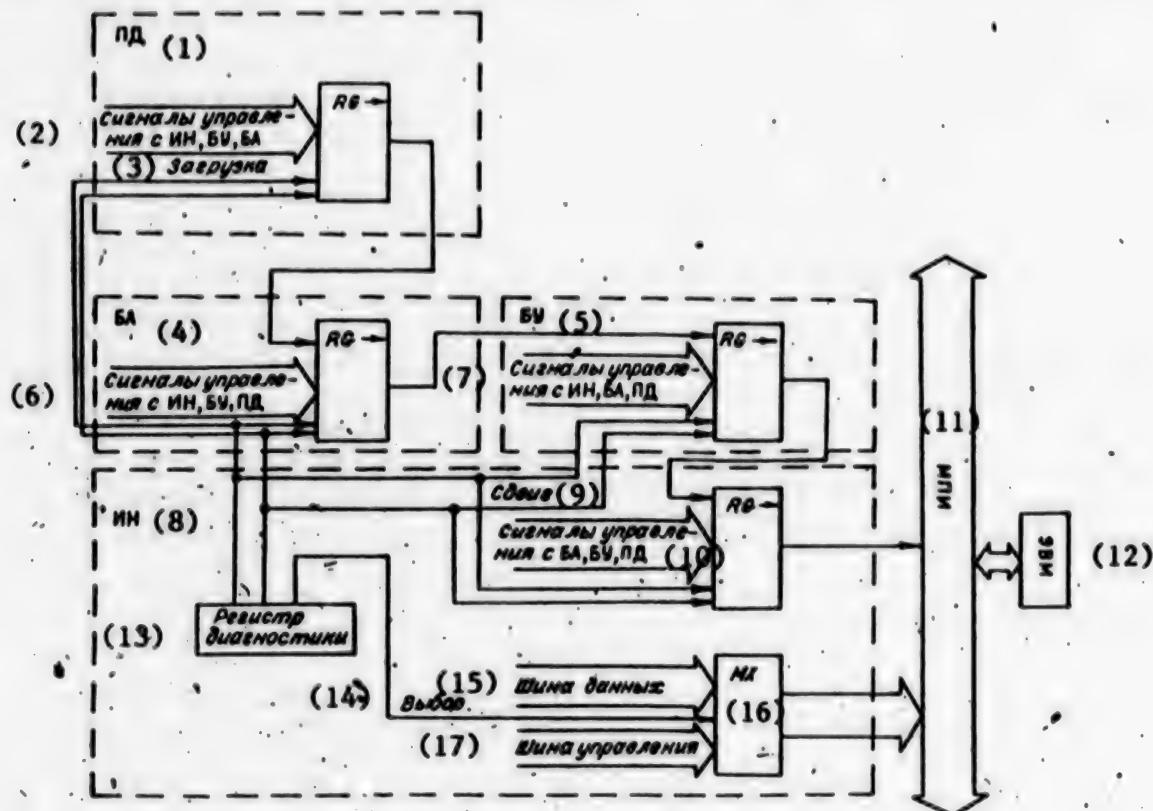


Figure 2. Block diagram of the high-speed peripheral processor's built-in diagnostics (RG = register).

- |  |   |
|--|---|
| 1--Data memory   | 8--Interface  |
| 2--Control signals from interface,<br>control unit and arithmetic unit | 9--Shift  |
| 3--Load  | 10--Control signals from arithmetic<br>unit, control unit and data memory |
| 4--Arithmetic unit   | 11--Parallel bus interface  |
| 5--Control unit  | 12--Control computer  |
| 6--Control signals from interface,<br>control unit and data memory     | 13--Diagnostic register   |
| 7--Control signals from interface,<br>arithmetic unit and data memory  | 14--Select  |
|  | 16--Multiplexer   |
|  | 17--Control bus   |

All control signals to the high-speed peripheral processor from the control computer pass via the special diagnostic register which allows the following functions to be carried out.

1. Stopping the high-speed processor's clock (shift to diagnostic mode).

2. Programming the clock to execute the required number of cycles.
3. Monitoring the clock's status (running, halted).
4. Reading the status of high-speed peripheral processor lines and busses into control computer memory for subsequent analysis.
5. Starting the high-speed processor's clock (exit diagnostic mode).

Built-in diagnostics are a significant aid in calibrating the high-speed peripheral processor even when used manually, the greatest effect, however, is achieved when the built-in diagnostics are supported by test software. In this case the volume of hardware required for built-in diagnostics does not exceed 3 percent of the Elektronika MS 1603 high-speed peripheral processor's total hardware.

The high-speed peripheral processor's diagnostic programs handle operational monitoring and monitoring by means of the built-in diagnostic hardware. If a fault is detected during execution of any high-speed peripheral processor operation, monitoring is stopped, the unit is shifted to the diagnostic mode and the check of the operation during which the error occurred is repeated with the assistance of the built-in diagnostic elements. The data received from the high-speed peripheral processor is compared with a reference. The comparison result is analyzed and a message is printed specifying the faulty component or block of components.

Since the use of built-in diagnostic resources, providing a greater scope of fault detection and isolation, markedly increases the time involved in testing, this occurs, as stated above, only when a fault is detected during operational monitoring. Thus, the built-in diagnostic resources do not come into play during error-free high-speed peripheral processor operational testing.

If only a general picture of high-speed peripheral processor operational capacity is required, the built-in diagnostic resources need not be used even if a fault is detected.

The testing of any high-speed peripheral processor function or element requires the following parameters to be specified:

1. The timepoint at which the check must be carried out (defining the number of cycles and half-cycles executed by the clock).
2. The signals subject to monitoring (depending on the order number corresponding to the sequence in which signals are "placed" at the shift register's outputs).
3. The reference values for monitored signals and information on the control bus and data bus.

The analysis of data received and the formation of a conclusion as to the fault source present the greatest difficulty in the use of built-in diagnostic resources. A strict arrangement of signal/fault source links is simpler when

one assumes that, in the case of an unreliable unit, several signals from a fault source will always be the same. A simple and highly reliable arrangement leads to the isolation of faults at the component block level and in many cases at the individual component or component group level.

Fault isolation at the component lead level is assured by the logical analysis of test results, in which a conclusion on the source of a fault is based not on the states of specific signals but on a combination of signal states. This type of analysis requires a significant degree of test program complexity and increased reference table sizes. At the same time, fault isolation at the component lead level is required only when calibrating blocks of components on special diagnostic equipment. It is redundant when testing the device as a whole.

The troubleshooting method described can be used successfully in the development of new peripheral processors and controllers and it allows a significant reduction in the costs of equipment calibration and repair incurred by computer manufacturers and users.

COPYRIGHT: Izdatel'stvo "Nauka", "Avtometriya", 1985

12746

CSO: 1863/355

UDC 681.142

## AN APPROACH TO IMPLEMENTING CAMAC MICROPROCESSOR MODULES

Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 2, Mar-Apr 1985  
(manuscript received 14 Dec 83) pp 106-108

[Article by Ye. M. Bazarnyy, A. N. Vystavkin, L. Z. Pososhenko, V. P. Reztsov and A. B. Furshchik under the rubric: "Automation of Scientific Research"]

[Text] CAMAC [computer automated measurement and control] hardware is widely used in the creation of systems for automating scientific research [1].

The use of microprocessors in CAMAC hardware allows substantial improvement in functional characteristics and increases in system efficiency.

In addition to the design of multicontroller microprocessor systems [2], a generation of "smart" modules needs to be developed to enable preliminary data processing directly in the control subprogram execution module and to perform a number of auxiliary tasks (automonitoring, parameter correction, etc.).

This article discusses one possible architecture for a new generation of CAMAC modules based on microprocessor components and points out the superior properties of these modules.

To increase efficiency in the preparation of design and program documentation and to increase product reliability it is reasonable for the microprocessor portion of the module to be isolated on a separate--microcontroller--board. This approach allows a unified series of modules to be produced.

The illustration shows a generalized block diagram of a "smart" CAMAC module which meets the requirements mentioned above.

The selection of the host LSIC [large-scale integrated circuit] is the critical point in module development. At this time, the authors consider the K580 series microprocessor unit to be suitable for this use. This selection is based on the following factors:

- The KR580IK80A microprocessor's throughput is adequate for the implementation of medium-scale control systems.

- The K580 series is rapidly being supplemented with various types of peripheral LSICs, thus allowing a wide variety of different modules at a high technical level.
- There is a great deal of software experience for the K580 series gathered from around the world to be drawn on.
- Possible incompatibility between the microprocessor controller's instruction set and that of the computer system is not important since the module has its own firmware and communicates with the crate bus through a fixed protocol independent of system commands.

An analysis of the control, acquisition, preliminary processing and operational information display tasks executed at the module level on a real-time basis allows the formulation of microcontroller requirements. It should have the following characteristics:

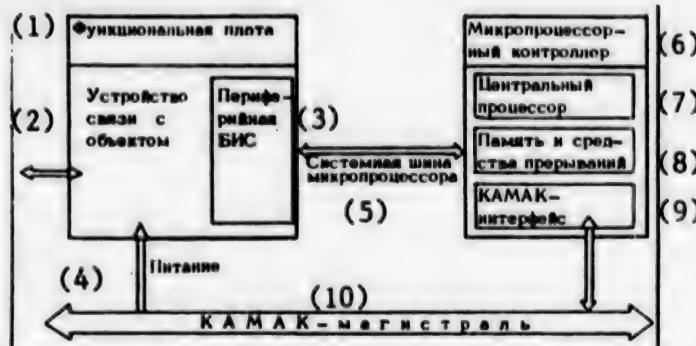
1. A speed of at least 500 thousand operations/second when executing register to register operations.
2. A main memory of at least 1K-byte.
3. A read-only memory (ROM) of at least 2K-bytes.
4. An 8-bit CAMAC bus input/output port.
5. The microprocessor system bus must contain:
  - 8 data bus lines
  - 16 address bus lines
  - 8 interrupt request lines with programmable priority levels
  - TTL Ø2 phase
  - Strobe outputs for read, write, input and output to open-collector registers
  - a RESET line
  - a direct memory access request
  - a hold request

The microprocessor system bus must be brought out to an interboard connector or an edge connector for connection to the function board. This type of bus allows a function board with various types of K580-series LSICs to be connected to the microprocessor with a minimum number of ancillaries.

A number of CAMAC modules along the lines of the approach described have been developed at the Special Design Bureau of the Academy of Sciences USSR Institute of Radio Engineering and Electronics.

The use of a microcontroller in a cassette tape interface module provided functions such as control of the tape transport mechanism and data read/write operations, conversion of data formats, retrieval of data on the tape, monitoring of recording accuracy and self-testing with minimal CAMAC bus and computer system occupation levels.

Microprocessor modules are rather complex devices and their use must be justified from the engineering and economic standpoints. At present there is no evaluation of the advantage of using microprocessor CAMAC modules in terms of computer utilization time and memory volume.



- 1—Function board  
 2—Device communicating with control objective  
 3—Peripheral LSIC  
 4—Power supply

- 5—Microprocessor system bus  
 6—Microprocessor controller  
 7—Central processor  
 8—Memory and interrupt handler  
 9—CAMAC interface

The guideline estimates given below define the quantitative characteristics of CAMAC microprocessor module efficiency under the following assumptions:

- A single-crate CAMAC system containing N microprocessor modules.
- The modules operate in an experimental real-time environment, so that the time for program execution by the microprocessor is assigned by this environment.
- The microprocessor programs are sufficiently developed and of adequate size (to 4K-bytes).
- The system computer/module operating protocol consists of task and data transmissions between the microprocessor's buffer and the computer. The microprocessor executes its assignments during the intervals between transmissions.
- The operating protocol is identical for all modules with microprocessors, although data content and the length of files transferred differ. All data is sent on R and W buses. Other bus signals are used only for module clocking by the system computer and these are reduced to a minimum.

Let us assume that the modules in a system carry out definite operations  $M_i$ ,  $i = 1, 2, \dots$ . We will call  $t_i$  the time for data and assignment transmission in operation  $M_i$ , and the time for the microprocessor to execute the  $i$ -th operation will be  $T_i$ . The total module-busy time ( $T_M$ ) is defined as:

$$T_M = \sum_i (t_i + T_i)$$

The time during which the system computer is occupied with crate bus communications is defined as:

$$T_{\text{com}} = \sum_i t_i$$

Autonomous execution of program operations by the modules has a favorable effect on computer occupation time and this is expressed by the ratio:

$$\eta_i = \frac{T_M}{T_{\text{com}}} = 1 + \frac{\sum_i T_i}{\sum_i t_i} = 1 + \frac{T_{\text{cp}}}{t_{\text{cp}}} .$$

in which  $T_{cp}$  is mean time for microprocessor module operation execution and  $t_{cp}$  is the mean time during which the system computer is occupied in data communications with the modules.

Operation execution time becomes greater when a module operates with electromechanical devices or when voluminous calculations are executed. Under these conditions the microprocessor program contains a set of iterative segments, such as flag request, iteration loops, pointer loops and nested loops. The execution of a 4K-byte program can occupy several hundred thousand microprocessor cycles. On the other hand, very concentrated data are transmitted on the CAMAC bus, thus their flow can be restricted and the data transmission rate can reach some tens of thousands of bytes per second. The analysis of specific tasks shows that system computer utilization times are typically reduced  $n_t$ -fold (from 10- to 100-times) in operations with microprocessor modules.

The use of microprocessor modules reduces the volume of programs in the system computer. This savings is achieved by moving part of the general software to microprocessor ROM, simplifying the executive program and reducing the number of transitions. The number of functionally identical object code programs for the KR580IK80A microprocessor is approx. three to four times greater than that in the popular SM-4 and "Elektronika-60" system computers. However, drivers proceed from the system computer to the individual modules via the crate controllers, while the built-in microprocessors interact directly with the function board.

As a result, one can assume as a guideline that a microprocessor module with a 4K-byte ROM reduces the size of the system computer program by approximately 1.5K-bytes. In our example the program volume advantage is:

$$\frac{t_p}{T_p} = \frac{V + 1.5N}{V} ,$$

in which V is the system control program volume in K-bytes and N is the number of microprocessor modules in the system.

Assuming  $N = 4$  and  $V = 8$ , we obtain  $\frac{t_p}{T_p} = 1.7$ .

The length of programs required for system computer operation with microprocessor modules can be reduced additionally through the use of a uniform data exchange protocols and a common CAMAC interface for all microprocessor modules. We will explain this in more detail.

A common data exchange protocol allows the generation of initial tables and protocols to be confined to a single program in system computer memory. An individual table modification program is then used for each module. A module's CAMAC interface is part of the microprocessor controller and identical in all modules. Thus, a common program driver can be used to transmit words. A reduction is achieved by structuring table data in such a manner as to facilitate machine processing using incremental and indexed addressing.

It is difficult to assess the degree to which the amount of software can be reduced by these factors; only operational experience will make this clear. The data shown indicate that microprocessor modules significantly reduce control program volume and system computer utilization time in operations running in an experimental environment. Thus, there is a possibility of increasing CAMAC system throughput within the framework of previous time values and of targeting basic efforts toward the processing of experimental data in order to increase their informational value.

#### BIBLIOGRAPHY

1. Nikityuk, N. M. "Programmno-upravlyayemye bloki v standarte KAMAK" [Program Controlled Elements in the CAMAC Standard]. Moscow: Energiya, 1977, 152 pp.
2. Kolpakov, I. F. "Organizatsiya programmno-modul'nykh sistem na osnove standartnykh interfeysov" [Organizing Modular Program Systems Based on Standard Interfaces]. PRIBORY I TEKHNIKA EKSPERIMENTA, 1979, No 2, p 7-33.

COPYRIGHT: Izdatel'stvo "Naukovo Dumka", "Upravlyayushchiye sistemy i mashiny", 1985

12746  
CSO: 1863/270

UDC 681.325

## COMPACT, HIGH-SPEED ALGORITHM FOR LAYING OUT PRINTED CIRCUIT BOARD RUNS

Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 2, Mar-Apr 1985  
(manuscript received 16 Jan 84, after review 7 Jun 84) pp 33-37

[Article by D. Ye. Zapolotskiy under the rubric: "Automation of Computer and System Design and Construction"]

[Text] The printed circuit connection layout algorithm described belongs to the class of primary algorithms [1-6] and was developed within the framework of an interactive system for designing two-sided printed circuit boards [7]. For this reason, algorithm speed was considered, *a priori*, as a requirement equally as important as the inherent demand for minimizing circuit run lengths and the number of junction openings. This resulted from the fact that, in order to provide psychological man/machine compatibility in the design process, real-time dialog during the layout phase is possible only within limited time frames (on the order of several seconds) for each circuit run.

The work was carried out for use on an ARM-R automated work site complex based on an SM-4 minicomputer with a 32K-word memory. This limited memory capacity heightened the demand for algorithm speed and also tightened data file structure and size requirements. Thus, the problem of providing a high-speed layout algorithm designed to run on a minicomputer with limited memory had to be solved under conditions of simultaneous and often conflicting requirements.

Before analyzing the layout algorithm's design logic we will describe the structure and organization of the data files. Their definition required considerable use of the properties of the layered orthogonal circuit run [1-4].

Let us examine the discrete-point construction space of a printed circuit board. We will compare each side of the board to a Boolean matrix [8] whose elements are matched with discrete points and have a value of 1 if the corresponding point is occupied. We will designate the Boolean matrix making up the side of the board on which vertical segments will be run the vertical segment matrix. The horizontal segment matrix will be the Boolean matrix for the side on which horizontal segments will be run. The requirement for the algorithm to run on a computer with a small memory capacity influenced the choice of representing printed circuit board fields as logic arrays. Since an SM-4 word can contain 16 data bits, the amount of memory required to represent

both sides of a two-sided printed circuit board can be reduced 16-fold in comparison to algorithms which require one machine word to describe a discrete point on a printed circuit board. Thus, the representation of two logic arrays for a 200 x 200-point printed circuit board requires approx. 5K-words instead of the usual 80K-words.

The use of Boolean matrices not only provides a compact form of representing construction space occupation data on the computer, it also allows the possibility of creating high-speed routines to determine free segment boundaries. This is achieved by means of a search for the boundary set of sequentially arranged null elements, located adjacent to the component which corresponds to any given point, in the (vertical segment or horizontal segment) matrices. The speed of the search is due to the fact that contiguous computer memory storage locations correspond to contiguous elements in logic array columns and this results in simultaneous processing of all bit groups.

A dual-stack unit [9] which can sort the value of any data elements regardless of their location in memory is used to isolate the free segment search regions. The values of the free segment boundaries are loaded into the stacks, or lists, in ascending coordinate order. Vertical segments are handled by the vertical segment boundary value list and horizontal segments by the horizontal segment boundary value list.

Thus, data describing the entire construction space of a two-sided printed circuit board and indicating all occupied discrete points are stored in the vertical and horizontal segment matrices, while data on each run plotted are organized as special lists of free segments. The lists (the vertical segment boundary value list and the horizontal segment boundary value list) are formed after a run connecting the entire set of contacts in a given circuit is plotted.

The algorithm's high-speed characteristics are established by effective procedures which facilitate the search for a free segment in a given region and within given boundaries in the field of a printed circuit board as described by the vertical and horizontal segment logic matrices, while the search region itself is defined by analyzing the sorted vertical or horizontal boundary value lists.

We will describe logic array matrix representation in the computer more accurately by assuming for simplicity that the size limit for these matrices is 200 x 200 discrete points. The vertical and horizontal segment matrices are then transformed into matrices with a 16-bit word dimension of 13 x 200. An overall conversion from working with matrices to working with vectors can be made if one considers the use of floating-point numbers with a 13-word mantissa whose order allows values over 200, for which a single word suffices completely. Consequently, a single 14-word digit is assigned per bit matrix line and the vertical segment matrix is represented by sequential bits corresponding to the vertical direction while the horizontal segment matrix is the horizontal direction. In this case, any X-point (in discrete points) will correspond to a bit in the Y position of the (X) vertical segment matrix mantissa (positions are counted from left to right) in an order equal to 0, or to a bit in the X position of the (Y) horizontal segment matrix mantissa. Only

a total of 5600 SM-4 memory locations are needed to store the vertical and horizontal segment matrices.

The search procedure for the free segments including a given X,Y-point (vertical in the vertical segment matrix and horizontal in the horizontal segment matrix), consists of locating the first non-zero bit in the position preceding Y for the (X) vertical segment matrix or X for the (Y) horizontal segment matrix and then determining the first non-zero bit in the next position. This operation allows the identification of the beginning and end values for a free segment containing the the given X,Y-point in the (X) vertical segment matrix or the (Y) horizontal segment matrix by normalizing all the digits defined above. First, all the bits preceding the coordinate sought, except for the nearest non-zero bit, are zeroed and the application of number-exponent normalization results in a value which defines the beginning of the segment. Zeroing of the preceding non-zero bit and repeated normalization will yield the end of the segment, the value of which is defined by the value of the exponent.

The operation used to find a free segment is as follows: OTR (X,Y,B,E,MAS); where OTR is the segment; X and Y are the coordinates of the point for which the segment containing that point is sought; B is the value of the beginning of the segment; E is the value of the end of the segment and MAS is the matrix identifier (vertical segment or horizontal segment matrix).

Operations to analyze free segments, the data on which are extracted from the vertical or horizontal segment matrices, are carried out separately for the vertical and horizontal segments using the applicable vertical or horizontal boundary segment value lists. Analysis of vertical segments is carried out to define their mutual location in order to identify a horizontal segment formation region. In this manner a segment tree is formed by alternately analyzing both sides.

During the process of segment incrementing, the special "Derevo" [Tree] procedure controls the precise moment of tree formation, for which the existence of a run in the given set of contacts is a necessary condition, and completes the process of forming new segments.

The inability to form a tree indicates the absence of a run and is determined upon reaching a permissible limit number of iterations or segments. The two-sided vertical and horizontal boundary segment value lists are the primary instrument for analyzing segments during the process of forming each run. The use of lists, or stacks, would seem to call for an increase in the memory requirement, but this is avoided by using the principle of independent run formation for each circuit since printed circuit board space occupation data are stored in the vertical and horizontal segment matrices which are permanently held in memory. Therefore, the computer memory must contain the current segment plotted at the moment of run layout and initial information on the circuits in the form of a set of contact coordinates. The resultant data on runs laid out can be effectively stored on disks. This arrangement is justified because times on the order of one second are required for processing data files during the formation of each run, while disk access involves milliseconds.

This trade-off assures access to lists as a high-speed instrument for analyzing the volume of unoccupied segments involved in each individual run. The following lists are used in the algorithm:

- 1) A list of circuit contact and intermediate point coordinates arising during run layout (SKON).
- 2) A list of vertical segment boundary values (SGVO).
- 3) A list of horizontal segment boundary values (SGCO).
- 4) A list for evaluating connectivity during tree formation (SD).
- 5) A dynamic list of segments with combined projections, horizontally sorted for the vertical segments and vertically sorted for the horizontal segments (SVG).

Three machine words are allocated to each list element. Therefore, the on-line memory required for storing these five lists is 1.5K-words on the condition that up to 100 segments are assigned to a run. This is adequate for a wide spectrum of runs. With this arrangement the segment volume limit can be increased easily without significantly increasing memory, which when combined with the portion allocated to the vertical and horizontal segment matrices (MVO and MGO respectively) amounts to some 7K-words.

By using the procedures listed and following the structural characteristics for data files as described above, a layout algorithm can be developed which allows the formation of a run in a forest of segments which in the initial stage are included in the set of coordinates for all circuit contacts.

The following designations will be used in describing the algorithm:

M—the total number of circuits

m—the current circuit number:  $1 \leq m \leq M$

L—the number of contacts in the current circuit

K—the total number of free segments forming the tree:  $K \geq L$

$i_1$ —the current point number on which the free segment  $1 \leq i_1 \leq K$  is defined

MAS=      MVO, if points on the vertical segment plane are analyzed  
              MGO, if points on the horizontal segment plane are analyzed

$P_1$ —a point with the coordinates  $i_1, j_1$ , in which the determining direction of segments in the MVO or MGO is linked to the  $i_1$  coordinate

$i_1 =$        $x_1$  for the MGO  
               $y_1$  for the MVO

$j_1 =$        $x_1$  for the MVO  
               $y_1$  for the MGO

$\min(P_1)$ —the value of the lower boundary (beginning) of the free segment formed for point  $P_1$

$\max(P_1)$ —the value of the upper boundary (end) of the free segment formed for point  $P_1$

It must be noted that the beginning value is incremented by 1 and the end value is decremented by 1. This is introduced in order to take into account manufacturing limits which do not allow the establishment of two junction openings at adjacent discrete points.

For simplicity we will assume that the coordinates of all contacts are located on one side, in the MVO for example, although this is not fundamental. Thus, the algorithm can be formulated in terms of the following instructions:

1. Start of circuit processing cycle:  $m = 1$ .
2. Start of single circuit processing cycle:  $K = L$ , MAS = MVO; the SKON  $\leftarrow i_1, j_1$  list is formed for  $l = 1, 2 \dots L$ ; the SGVO  $\leftarrow \min(P_1), \max(P_1)$  list is formed.

Five segments are illustrated as an example in Figure 1. Here, the dashed lines indicate segment boundary values sequence in the SGVO:  $\min(P_1), \min(P_3), \min(P_5), \min(P_4), \max(P_1), \max(P_3), \max(P_5), \max(P_4), \min(P_2), \max(P_2)$ ; jump to 4.

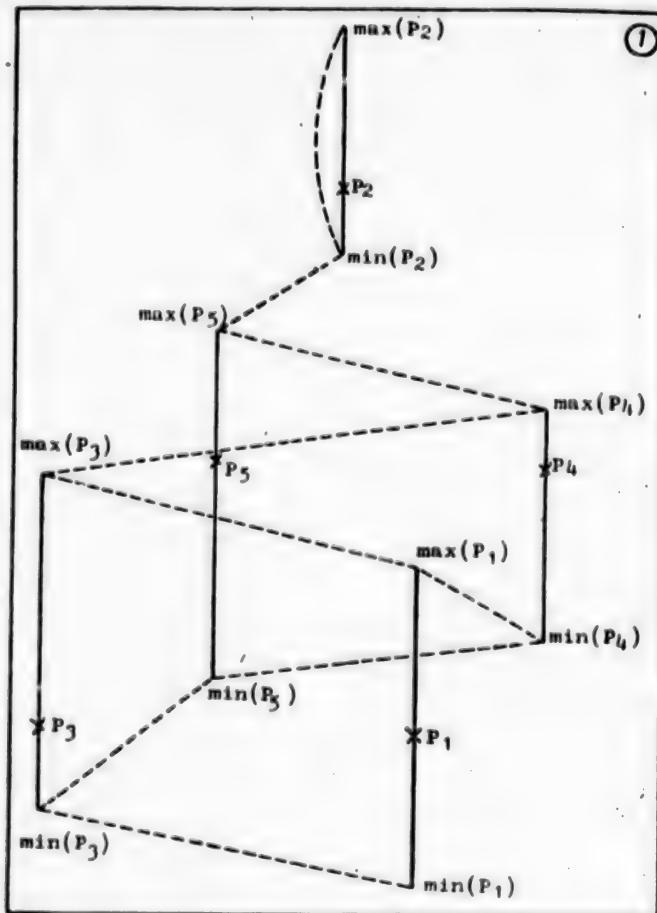


Figure 1. Representation of free segment boundaries in the list.

3. If  $MAS = MVO$ , then  $MAS: = MGO$ ; if  $MAS = MGO$ , then  $MAS: = MVO$ , i.e. a change of board plane takes place for analysis.
4. Depending on the MAS value, either the SGVO or SGG<sub>U</sub> is set up. An index is created of the point with the minimum value of the lower boundary.
5. Movement takes place toward the larger values in the list. The n index of the current boundary is defined.
6. If the maximum value for the upper boundary is encountered, jump to 13
7. If the list value is that of free segment upper boundary, jump to 9.
8. Current lower boundary analysis module:  $i_{min} = \min(P_n)$ ; the  $j_n$  coordinate is transferred to the SVG. There only the segments with lower boundary values less than or equal to  $i_{min}$  and upper boundary values greater than  $i_{min}$  are sorted by  $j$  coordinate. Jump to 5.
9. Start of the current upper boundary analysis cycle:  $i_{max} = \max(P_n)$ . By this time the segment with a common projection region on the axis of coordinate  $i$ :  $i_{min} \leq i \leq i_{max}$  is in the SVG. If the SVG segments are connected, and this is verified by the SD, jump to 12.

Figure 2a shows a fragment linked to the detection of the end of a free segment which includes point  $P_1$ . Now the SVG contains  $j$  values in the following order:  $j_3, j_5, j_1, j_4$ , and  $i_{min} = \min(P_4)$  and  $i_{max} = \max(P_1)$ .

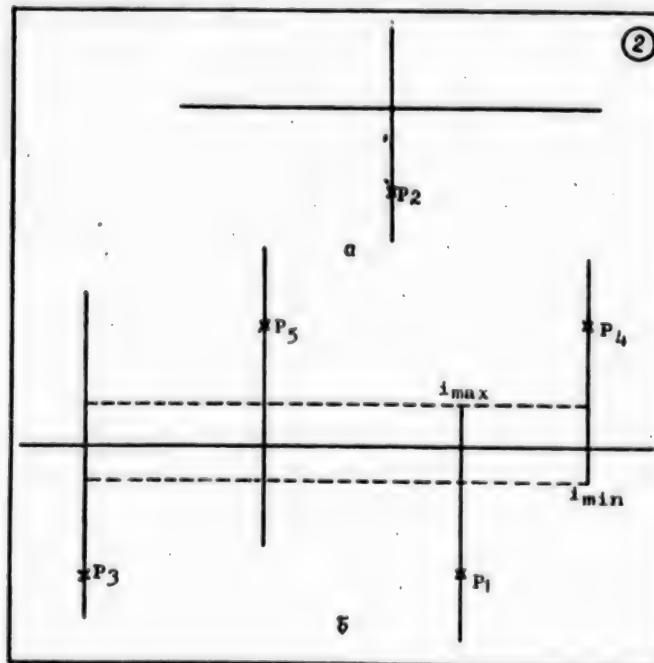


Figure 2. Definition of the initial free segment intersection region and orthogonal segment thus formed (b); orthogonal segment formation for one initial free segment (a).

Figure 2b illustrates a boundary condition in which the SVG contains only one segment with  $i_{\min} = \min(P_2)$  and  $i_{\max} = \max(P_2)$ .

The SVG is used to determine the minimum  $j_{\min}$  and maximum  $j_{\max}$  values of coordinate  $j$ . Thus, when the SVG contains only one segment the following boundary values are assumed:  $j_{\min} = 1$  and  $j_{\max} = N$ , where  $N$  is the maximum dimension expressed in discrete points.

The new segment index is formed:  $k = k + 1$ . A new segment is formed in the plane corresponding to  $MAS1 = MGO$ , if  $MAS = MVO$ , and  $MAS1 = MVO$ , if  $MAS = MGO$ . To ensure that the free segment sought intersects the entire set of segments, a search is executed in the SVG for a point  $P_k$  whose coordinates, together with the boundaries of the segment including point  $P_k$ , must insofar as possible satisfy the conditions:

$$\begin{aligned} i_k &= i_n, \\ i_{\min} &\leq i_k \leq i_{\max}, \\ \min(P_k) &\leq j_{\min}, \\ \max(P_k) &\geq j_{\max}. \end{aligned}$$

The OTR operation is used for the segment search.

10. If a segment satisfying the indicated requirements (see Fig. 2a) is found, then jump to 12.
11. A search is carried out for the longest segment which intersects the segment whose upper boundary caused the jump to 9 (see Fig. 2b). Other heuristic solutions could also be used here.
12. End of the upper boundary analysis cycle. The  $j_n$  coordinate is released from the SVG. The  $i_k$  and  $j_k$  coordinates, as well as the  $\min(P_k)$  and  $\max(P_k)$  boundaries are transferred to the applicable lists. Segment connectivity is determined and latched in the SD. Jump to 5.
13. A group of operations identical to those in 5-12 are executed in inverse order (i.e. from the higher to the lower boundary values). This is done in order to isolate all possible regions of combined free segment projections along axis  $i$  without their exhaustion. In this group of operations an orthogonal segment is formed after the determination of the lower boundary values for the segments examined and the SVG is loaded only after the sequence of upper boundary values is discovered.
14. If a tree is formed, jump to 17.
15. If the number of junctions on the different sides of the board has not exceeded the maximum permissible limit, jump to 3.
16. Issue a message that a run was not found.
17. Formation of a run.

18. End of the single-circuit processing cycle:  $m=m+ql$ . If  $m \leq M$ , jump to 2.

19. End.

A group of FORTRAN programs with indications of good speed in laying out circuit runs (construction space discrete-point dimension of 1.25 mm) has been developed using this algorithm as a basis. Less than one second is required for run plotting and this time is not strictly dependent on the number of contacts in the circuit. This loose dependence on the number of connected points is due to the fact that free segment formation begins all at once from all circuit contact coordinates and that, during the orthogonal intersecting segment search, the point which assures the connection of the maximum possible number of initial segments is identified. It is obvious that the time characteristics are basically tied to the size of the segment search region and not to the quantity of orthogonal segments in that region.

In conclusion, it can be said that at its high layout speed the algorithm provides not less than 95 percent of the run layouts for saturated boards (on the order of 50-60 microcircuits and 180-200 circuits on a standard 170x200-mm<sup>2</sup> board). The run plotting program package is included in an interactive printed circuit board system [7] together with a component arrangement package [10] and a graphics editing system on the EPG-400 [11] which supports a real-time interactive environment. The high quality characteristics and real-time interactive implementation during component arrangement and run layout allow machine execution levels of up to 100% in high-density printed circuit boards with the use of the algorithm described above.

#### BIBLIOGRAPHY

1. Selyutin, V. A. "Mashinnoye konstruirovaniye elektronnykh ustroystv" [Machine Design of Electronic Devices]. Moscow: Sov. radio, 1977. 384 pp.
2. Loshakov, V. N. "Sistemy avtomatizatsii proyektirovaniya BIS s primeneniem EVM" [Computer-Aided Systems for Automating Large-Scale Integrated Circuit Design]. ELEKTRON. PROMYSHLENNOST', 1970, No 2, p 45-48.
3. Nogis, R. V. and Yurkunas, D. Yu. "Algoritm trassirovki pechatnykh soyedineniy s malym chislom poverotov" [Algorithm for Laying Out Printed Circuit Boards with a Low Number of Turns]. In book: "Vychislitel'naya tekhnika" [Computer Technology]. Kanaus: KPI, 1970 v 1, p 326-331.
4. Andreyev, G. D., Petukhov, G. A. and Skorubskiy, V. N. "Kompensiruyushchiy algoritm poiska malopovorotnykh putey" [Compensatory Algorithm for Seeking Paths with a Low Number of Turns]. ibid, 1972, v 3, p 380-386.

5. Gurevich, D. Z. and Selyutin, V. A. "Algoritmicheskiye metody proyektirovaniya topologii BIS yacheychnogo tipa" [Algorithmic Methods for Designing Cell-Type Large-Scale Integrated Circuit Topology]. In book: "Metody rascheta i avtomatizatsii proyektirovaniya mikroelektronnykh TsVM" [Methods for Evaluating and Automating the Design of Microelectronic Digital Computers]. Kiev: IK AN USSR, 1973, p 83-92.
6. Shteyn, M. Ye. and Shteyn, B. Ye. "Metody mashinnogo proyektirovaniya tsifrovoy apparatury" [Machine Design Methods for Digital Equipment]. Moscow, Sov. radio, 1983. 296 pp.
7. "Dialogovaya sistema proyektirovaniya pechatnykh plat na komplekse ARM/A" [Dialog System for Designing Printed Circuit Boards at an "A" Automated Work Site Complex]. A. Ya. Vol'fenzon, G. P. Demidov, D. Ye. Zapolotskiy et al. USiM, 1983, No 2, p 32-36.
8. "Avtomatizatsiya proyektirovaniya pechatnykh blokov s modulyami proizvol'noy formy" [Automation of the Design of Printed Circuit Elements with Random-Form Modules]. Ye. P. Gerasimenko, V. I. Kot, I. Ya. Landau and V. M. Somkin. M: Mashinostroyeniye, 1979. 168 pp.
9. Knut, D. "Isskustvo programmirovaniya dlya EVM" [The Art of Computer Programming]. Moscow: Mir, 1976, v 1, 736 pp.
10. Zapolotskiy, D. Ye. and Vol'fenzon, A. Ya. "Algoritm razmeshcheniya odnogabaritnykh elementov metodom potentsialov" [Algorithm for Laying Out Single-Dimension Elements By Means of the Potential Method]. USiM, 1983, No 5, p 32-34.
11. Demidov, G. P., Peskov, V. and Shteyman, D. M. "Interaktivnaya graficheskaya sistema na base mini-EVM" [Minicomputer-Based Interactive Graphics System]. ibid, p 87-89.

COPYRIGHT: Izdatel'stvo "Naukovo Dumka", "Upravlyayushchiye sistemy i mashiny", 1985

12746  
CSO: 1863/270

JPRS-UCC-85-006  
5 September 1985

UDC 681.7

#### OPERATING ALGORITHM FOR ADAPTIVE TRANSMISSION SYSTEM

Novosibirsk AVTOMETRIYA in Russian No 2, Mar-Apr 85  
(manuscript received 4 Nov 82) pp 69-74

ALEKSANDROV, A. B. and DOLOTIN, Yu. G., Vladimir

[Abstract] A basic feature of transmission and reeception of optical signals in a free atmosphere is the effect of turbulence in refraction. The present article reports on analysis of a simple transmission system and development of an algorithm to describe its functioning. The authors rejected systems based on non-linear optics and worked with a shaped mirror as the adaptive element of the system, and multi-channel phase modulation as the principle for compensation for turbulent distortions. Actual analysis in the study was limited to an adaptive system that measured a wave front characterized by an inert adaptive element and stable focal length. The delivered signal intensity was a further determinant of operational effectiveness. Algorithms for assessing phase values and for evaluating the adaptive system are presented and discussed. Analysis of the newly synthesized algorithm made it possible to determine the average intensity of a signal striking a target in a large number of adaptations. Figures 2; formulas 27; references 4: 2 Russian, 2 Western.

[354-12131]

UDC: 681.327

#### INTERFACE BETWEEN Yes COMPUTERS, SENSORS AND ANALOG SIGNAL RECORDERS

Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan 85  
(manuscript received 17 Apr 84) pp 101-102

BONDOKSIY, S. V., DEYEV, V. V., TITOV, M. S. and SHVED, V. G.

[Abstract] The series-produced YeS computers do not include hardware to organize direct input and output of analog signals with the use of intermediate converters. The device described in this article allows simple direct input-output of continuous random processes by means of the YeS

computer direct control devices, which function independently of the input-output devices, since they are intended to support communications among processors. The device includes an analog-digital and digital-analog converter and a unit to interface these devices with the direct control devices. The interface unit includes an input-output controller plus input and output circuits. The controller organizes requests for information input and output and synchronizes the operation of the device as whole. It includes a clock and request formation circuits. The input and output sections both include buffers for temporary information storage. The input-output software was written in Assembler and can operate either independently or under the control of the operating system, though this requires special OS modules to support correct performance of privileged data read and write instructions as well as external interrupt handlers. The device has been used for input and output of speech signals as well as underwater acoustical noise at 10-20 Kbytes per second. It has been connected to YeS-1022 and YeS-1035 computers. The interface module consists of 10 series 155 integrated circuits. Figure 1; references 3 Russian.

[196-6508]

SOFTWARE

UDC 681.3.06:681.326.7

## CLASSIFICATION OF REAL-TIME PROGRAM TESTING TASKS AND METHODS

Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 2, Mar-Apr 1985  
(manuscript received 5 Apr 83, after review 23 Oct 84) pp 57-62

[Article by I. N. Dolganyuk and Ye. Ya. Karpovskiy under the rubric: "General Control System Software"]

[Text] The testing of real-time programs, one of the most important steps in program debugging, is a critical problem in modern programming theory. Usually debugging is understood as the group of activities involved in detecting (identifying) and correcting errors in programs. These activities are grouped in stages: establishing the presence of an error, identifying the error and, finally, correcting the error.

In the course of debugging programs, developers are faced with a number of questions involving the determination of current test method capabilities and suitability. Similar, albeit more critical, tasks must be resolved by program end users since in spite of the large number of works published recently there is no single methodological approach to the problems of program testing and debugging. The various approaches and methods examined in the known literature only emphasize the absence of an overall theory of testing, as correctly noted in [1]. Furthermore, there is no single opinion in the literature on just what should be included under the heading of program testing. On one hand, [2] considers testing to be a method of checking program accuracy, with program accuracy being understood as the degree to which a program satisfies the formalized demands (program specifications) imposed on it. From this definition it follows that testing should result in an indication of a program's accuracy or inaccuracy. On the other hand, some authors [1,3] have rather convincingly pointed out that testing is not capable of establishing program accuracy since it only permits a judgement as to the existence of errors [1] and can be used to prove the presence of errors [3], but not their absence.

An understanding of the nature of the testing process is essential in classifying the methods and tasks involved in seeking errors in programs. Therefore, we will note that since the strategic goal (establishing program accuracy) cannot be achieved, a set of tactical goals should be set up to answer the following questions during testing:

What to test?

How to establish the existence of an error?

How to test?

When to end testing?

Thus, testing takes the form of a dynamic process with changing tactical goals in which a cutoff decision depends on a certain general principle of decision-making on the basis of criteria which allow a judgement to be made on the degree to which set goals have been achieved.

With this in mind, we will employ the following statement as a working definition. Testing is a dynamic process whose strategic goal is establishing the presence of an error in a program by means of the following sequence of actions:

1. Test objective definition.
2. Tactical goal assignment.
3. Criteria selection.
4. Test design and generation.
5. Test introduction and program execution.
6. Program result processing to determine the existence of errors.
7. Testing cutoff point determination.

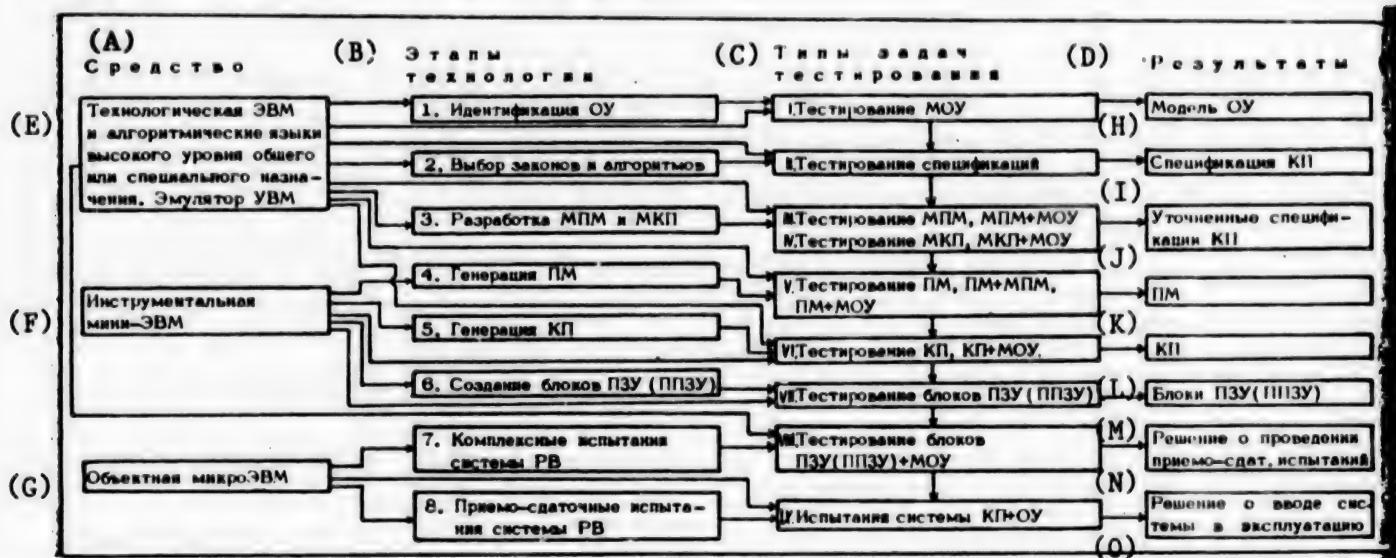
Modern real-time microprocessor-based control systems are characterized by a rather complex programming process whose steps include identification of the control objective, selection of control principles and algorithms, development of algorithmic models of program modules in high-level languages, cross-assembly of program module models and generation of program packages, preparation of a read-only memory (ROM) board (writing the program package to programmable read-only memory (PROM) and, finally, adjustment of the real-time control system by changing ROM units (MICON-IV system), alphanumeric keystrokes (TDC 2000) or entering problem-oriented instructions via the technician's console (UDAC "Eptak" system).

The complexity and multi-layer nature of the real-time program development process determine the variety of testing tasks and resources (see the chart) currently employed by microprocessor system developers and users.

The process of developing program packages for microprocessor systems operating in a real-time mode usually begins with the identification and creation of control objective models on a general-purpose computer. Control objective model creation and the subsequent selection of control principles and algorithms (e.g. the analytical formulation of a regulator) result in a program package specification used as the basis for developing simulation models of program modules and a model of the program package by means of modeling (emulation) software on a general-purpose (engineering) computer. In this process, the general-purpose computer is used to perform program module model and program package model/control objective model\* testing for open- and closed-type control systems (see tasks I, II, III, and IV in the chart).

\* Hereinafter the "/" sign indicates a link between system elements.

During a given stage, the criteria for establishing an error are, for example, the values of control (regulation) quality indicators in a closed program package/control objective model system.



Possible arrangement of an automated process for creating real-time microprocessor system program packages.

A—Resource

B—Process stages

C—Testing tasks

D—Results

E—Engineering computer and general-purpose or specialized high-level algorithmic languages. Control computer emulator.

F—Developmental minicomputer

G—Target microcomputer

1. Control objective (OU) identification
2. Rule and algorithm selection
3. Program module model (MPM) and program package model (MKP) development
4. Program module (PM) generation
5. Program package (KP) generation
6. ROM (PROM) development
7. Real-time system overall testing
8. Real-time system acceptance testing

- I. Control objective model (MOU) testing
- II. Specification testing
- III. MPM and MPM/MOU testing
- IV. MKP and MKP/MOU testing
- V. PM, PM/MPM and PM/MOU testing
- VI. KP and KP/MOU testing
- VII. ROM (PROM) testing
- VIII. ROM(PROM)/MOU testing
- IX. KP/OU testing\*

H—OU model

I—KP specification

J—Refined KP specification

K—PM

L—KP

M—ROMs (PROMs)

N—Decision to carry out  
acceptance testing

O—Decision to make system  
operational

Program module and program package model testing results in refined program package specifications used by a tooling minicomputer to generate program modules and program packages for microprocessor systems based on the target microcomputers. A combination of a general-purpose computer, emulating the control objective, and a tooling minicomputer, executing the program modules and packages, can be used to test the program modules and packages generated. During this stage, the program modules and program packages are debugged according to the results of program module, program module/program module model, program module/control objective module and program package/control objective model testing (see tasks V and VI on the chart). The testing cutoff point can be determined by achieving closed control system program package and control objective model quality indicator values and/or by evaluating the production loss factor  $U_p$  [33].

Microprocessor-controlled systems often do not permit program package debugging on the control objectives themselves. For this reason, program package/control objective model system debugging can in many cases be considered as general and final. Debugging on the tooling minicomputer results in the production of ROM (PROM) units for the target microcomputer. ROM (PROM) units are tested during the acceptance trials which serve as the basis for the decision to make the microprocessor system operative in the target microcomputers.

The specificity of the production process for real-time microcomputer system program packages (see the chart) gives rise to new testing tasks not covered in prior classifications as applied to real-time program testing problems based on our assumption of testing as a dynamic process (Table 1).

As Table 1 shows, the testing process can be described as a set of seven groups of classification tags  $\langle P_1, \dots, P_7 \rangle$ , where each of the groups is a matrix line:

$$P_i = [P_i^{(1)}, P_i^{(2)}, \dots, P_i^{(k_i)}], \quad i = 1, \dots, 7; \quad 3 \leq k_i \leq 14$$

Next, a possible variant of the testing statement can be obtained using a morphological analysis method if one of the elements in each matrix line is isolated and all of these isolated elements are connected by lines.

When practical implementation possibilities for the statements resulting from morphological analysis are not taken into consideration, a total of  $\prod_{i=1}^7 k_i = 258720$  can be formed. Table 2 shows the classification tags for statements examined in the known literature. Table 2, more or less a guide to published literature, shows that some 70 statements have been examined to date. These, however, are not uniformly distributed according to the various stages in the real-time microprocessor program package development process.

An analysis of Table 2 shows that the program module, program module/program module model, program module/control objective model, program package and program package/control objective model testing tasks involved in program module and program package generation are the most frequently examined.

Table 1. Real-time program testing tag classification

Classification tags	Source in which tags are examined
Test objective ( $P_1$ )	
1. MOU*	
2. Control program specification*	
3. Program module (including PM/MPM and PM/MOU)	[2-5; 8-22]
4. MPM, MPM/MOU*	[23; 24]
5. MPM/control computer emulator	[25; 26]
6. MKP	[5; 20; 22; 23; 27; 28]
7. MKP/control computer emulator	[3; 5; 26; 29]
8. MKP/MOU*	
9. MKP/MOU/control computer emulator*	
10. KP	[1-5; 7; 8; 10; 12; 17; 21; 30-37]
11. KP/MOU	[6; 18]
12. ROMs (PROMs) with KP*	
13. ROMs (PROMs) with KP/MOU	[5; 38-41]
14. Control system (ROMs [PROMs]/control computer/OU	[5; 6; 24]
Tactical goal of testing ( $P_2$ )	
1. Check functions executed by program*	[8; 20; 24; 32; 38; 39; 41-43]
2. Check program logic structure (peaks, segments, etc.)*	[1-9; 18; 21; 25; 26; 28; 34; 36; 37; 42-44]
3. Test program under extreme operational conditions*	[1; 3; 6; 10; 25; 26; 28; 32; 38-41]
4. Check program operation timing modes (including process priorities, real-time event synchronization, etc.)*	[6-10; 11; 18; 21; 23-26; 29; 38; 39; 41]
Criteria for establishing the existence of an error ( $P_3$ )	
1. Equivalence between specifications and program operating results (determinate case)	[38]
2. Equivalence between specifications and program operating results (stochastic case)*	{ }
3. Correlation for determining improper operation of an individual program branch	[6]
4. Condition (decision) tables	[19; 24]
5. Program operation time using test data file	[15]
6. Other checks relative to the test subject under examination	[7; 11; 15; 20; 26; 29; 41]
Testing cutoff criteria ( $P_4$ )	
1. Economic criteria	[2; 3; 5-7; 10; 17; 23; 26; 31; 33]
2. Criteria based on probabilistic and statistical models of program reliability	[3; 5-7; 16; 27; 35; 36; 45-49]
3. Other probabilistic and statistical criteria	[2; 5; 6; 9; 11; 28; 40; 50-52]

Table 1. (cont.)

Classification tags	Source in which tags are examined
Testing cutoff criteria ( $P_4$ ) (cont.)	
4. Structural criteria (testing completeness)	[2; 4-8; 13; 32; 36; 48]
Test design (generation) resources ( $P_5$ )	
1. By reduction of program path complexity	[2; 4; 5; 14]
2. By reduction of program path execution time	[2; 4; 5; 10; 28; 41]
3. By reduction of program path completion	[2; 4; 5; 28]
4. Step-wise testing of program package (top-down or bottom-up)	[3; 5; 6; 27; 37; 46]
5. Symbolic program execution (testing)	[8; 13; 15; 22; 36; 42; 53]
6. Generation of boundary and extreme values for input data	[3; 6; 8; 13]
7. Flowchart method	[3; 7]
8. Generation of random tests with a given dispersion rule	[5-7; 9; 13; 21; 22; 25-27; 50]
9. Test generation by means of active, multi-factor experimental methods	[34]
10. Insertion of control objective modeling results at program input	[5-7; 10; 12; 17; 18; 20; 21; 23; 26; 38; 39; 41; 54]
11. Creation of data bases with actual control objective operational results	[6; 24; 38]
Means of inserting tests at program input ( $P_6$ )	
1. Determinate	[2; 3-6; 10; 13-15; 39]
2. Stochastic	[3; 4; 5; 7; 10; 13; 21; 26; 27; 38; 39; 41; 50]
3. Mixed*	[3; 6; 24]
Methods of processing test results to establish the existence of errors ( $P_7$ )	
1. Method of defining situations for testing	[19]
2. Use of decision tables	[41]
3. Calibration program method	[23]
4. Boundary sampling method	[24; 27; 38]
5. Random search method with autoinstruction based on automaton probability theory	[11]
6. Other methods	[3-5; 15; 18; 25; 43]
7. Methods for processing the results of active, multifactor experiments*	[34]

- Notes: 1) An asterisk indicates classification tags which are not covered in sufficient detail in the literature known to the authors.
- 2) The testing cutoff criteria (determination of the moment at which to end program testing) can be interrelated or can change dynamically as the test goals change; the choice of these criteria can be ambiguous. This same observation applies to the other classification tags.

Table 2. Classification tags for tasks published in the literature known to the authors

(1) Типы за- дач тес- тирова- ния	Классификационные признаки (2)							(3) Источник, где описана задача тестирования
	$P_1$	$P_2$	$P_3$	$P_4$	$P_5$	$P_6$	$P_7$	
III	4	3	—	1	10	—	3	[23]
		4						
	4	1	4	—	11	—	4	[24]
		3						
III, IV	5	1	—	—	—	—	6	[25]
	7	2	—	—	—	—		
		3						
	4							
IV	5	1	5	1	8	2	—	[26]
	7	2			10			
		4						
	6	1	—	2	4	2	4	[27; 47]
V	6	3	—	1	10	—	3	[23]
		4						
	6	2	—	3	2	—	—	[28]
		3			3			
VI	7	4	5	—	—	—	—	[29]
	3	1	—	3	1	1	6	[2; 4; 5; 10]
		3			2			
VI, IX		4			3			
	3	2	3	4	6	1	—	[6]
		3	1	5	3	—	5	[11]
	3	1	6	—	5	2	—	[20—22]
VIII	10	1	—	1	4	1	6	[2; 4; 5; 10]
		3			10			
		4						
	10	2	—	1	9	3	7	[33; 34]
IX	11	1	3	1	8	3	—	[6]
		4		3	10			
				4	11			
	13	1	1	3	8	1	4	[38—40]
IX		3			10	2		
		4						
	13	1	6	—	2	2	2	[41]
		3			10			
IX	14	1	3	—	11	—	4	[27]
		3						
		4						

- (1) Testing task types
- (2) Classification tags
- (3) Source describing testing task

Notes: 1) Classification tags defined in Table 1.  
 2) Dashes indicate classification tags not defined in the task examined.

The tasks with classification tags  $P_1^{(1)}$  and  $P_1^{(2)}$  are insufficiently covered in the literature on software testing. However, note must be made of works such as [55] which identifies control objectives in detail, i.e. substantively, and covers the problem of testing control objective models. The testing of specifications relates in our opinion to the task of proving program accuracy, examined in works such as [56]. Tasks designated by  $P_1^{(1)}$  and  $P_1^{(2)}$  are an inherent part of the process of developing real-time systems; their solution in terms of a general theory of testing will open up the possibility of developing highly reliable real-time systems.

The first two stages of the process (see the chart) are followed by the more interesting and vital (in the sense of subsequent decision-making) third stage which includes type III and IV tests, subjects not adequately covered in the known literature (see Table 2).

The process of developing program packages for microcomputers must take into consideration system users as well as developers. This has not always been the case (see classification tag  $P_4^{(1)}$  in Table 2). As the real-time microprocessor system production process becomes more complicated, a demand has arisen to consider end-user interests during the application of new test methods, as in the final analysis these users are reflected in economic terms. In particular, third stage statements differing from those in Table 2 are appearing. For example:

$$\langle P_1^{(4)}, P_2^{(2)}, P_3^{(2)}, P_4^{(1)}, P_5^{(9)}, P_6^{(3)}, P_7^{(7)} \rangle.$$

Such a statement is obviously more responsive to user interests and differs from type III tasks developed earlier which are more in line with the interests of developers. Thus, a program module model in the real-time system production process (see the chart) consists of emulating that module on the tooling computer [23,24]. During the testing of program operating logic the user is often interested in testing a module's logic structure, usually in the form of a plot model of control transmission in the program. The determinate approach to selecting criteria for establishing the presence of errors does not usually satisfy the user since it does not provide a picture of actual program operation. For this reason a stochastic approach to establishing errors was chosen for the statement of the type III task discussed. An increasing number of authors (e.g. [2, 4-6, 10, 23, 26]) now recognize the need to relate the testing cutoff point to economic feasibility, but there are still approaches with limits, since primarily the interests of developers are considered.

Conditions for testing program module models can be examined as a series of experiments, and tests can be generated using a mathematical theory of experimentation whose utilization must be different from that in [34] where this theory is used for acceptance testing of finished software. The mathematical device in this theory simplifies the processing of test results. In selecting a means for inserting tests at the program module model input one can see that the determinate method does not correspond with actual program operation, in which data inputs are random. At the same time, the stochastic test input method is a passive experiment. The mixed method responds best to the demands of an active multifactor experiment. The need for developing a

method for testing graphic program modules based on a mathematical theory of experimentation arises from the type III task statement put forth. User interests can be reflected by an economic criterion for test completion with allowance made for possible losses due to running programs not adequately debugged.

Type IV tasks should be considered the most complex and critical at this time. These involve test objects ( $P_1^{(8)}$  and  $P_1^{(9)}$ ) which consist of a complex of program package/control objective models and program package/control objective models/control computer emulator, which in turn are the basis for generating program packages in subsequent production stages.

Task statements which are rather well covered in the literature arise in production stages 4, 5, 7 and 8 (see the chart). However, each of these is usually a local statement and has its own solution. Therefore, it is appropriate to further define known type V, VI, VIII and IX tasks in the context of an overall approach to testing at all stages of real-time program package production.

Type VII is an exception to these tasks. In many control systems this task cannot be implemented due to high hardware costs. Here a sort of synthesis of hardware and software testing takes place.

An analysis of tables 1 and 2 shows that practical methods for resolving local testing tasks arising during the individual stages of the production process (see the chart) are taking precedence over the development of general theory and that current theoretical research is concentrated primarily on type IV, V and VIII tasks. The abundance of formal task statements arising in a morphological analysis of Table 1 can lead one to the pessimistic conclusion that a search for general principles in program package testing processes is impossible. Therefore, at this stage the problems of developing a common theory of testing and establishing a unified mathematical device to resolve testing tasks at all stages of production are of critical importance. To this end, a synthesis of scientific trends in the formal demonstration of program accuracy and the use of a mathematical theory of experiments for the testing of real-time programs is promising in our opinion.

#### BIBLIOGRAPHY

1. Shurakov, V. V. "Nadezhnost' programmnogo obespecheniya sistem obrabotki dannykh" [Data Processing System Software Reliability]. Moscow: Statistika, 1981. 216 pp.
2. Lipayev, V. V., Pozin, B. A. and Blau, S. A. "Analiz strategiy testirovaniya logiki programm" [Analysis of Program Logic Testing Strategies]. KIBERNETIKA, 1982, No. 2, p 45-66.
3. Mayers, G. D. "Iskusstvo testirovaniya programm" [The Art of Program Testing]. Moscow: Finansy i statistika, 1982. 176 pp.

4. Lipayev, V. V. "Nadezhnost' programmnogo obespecheniya ASU" [Automated Control System Software Reliability]. Moscow: Energoizdat, 1981, 240 pp.
5. Lipayev, V. V. "Kachestvo programmnogo obespecheniya" [Software Quality]. Moscow: Finansy i statistika, 1983, 264 pp.
6. Teyyer, T., Lipov, M. and Nel'son E. "Nadezhnost' programmnogo obespecheniya" [Software Reliability]. Moscow: Mir, 1981, 323 pp.
7. Glass, R. "Rukovodstvo po nadezhnomu programmirovaniyu" [Handbook for Reliable Programming]. Moscow: Financy i statistika, 1982, 256 pp.
8. Borzov, Yu. V. "Metody testirovaniya i otladki programm EVM" [Computer Program Testing and Debugging Methods]. Riga: Izd-vo Latv. un-ta, 1980, 88 pp.
9. Gordiyenko, A. V. "K voprosu effektivnosti kriteriya testirovaniya programm po putym" [Effectiveness of the Program Path Testing Criterion]. USiM, 1982, No 2, p 107-108.
10. Lipayev, V. V. "Avtomatizatsiya proyektirovaniya programmnogo obespecheniya dlya upravlyayushchikh sistem" [Control System Software Design Automation]. AVTOMATIKA I TELEMEKHANIKA, 1982, No 2, p 92-104.
11. Dymarskiy, Ya. S., Kudinov, A. M. and Pustobayeva, L. A. "Ob odnom metode testirovaniya vychislitel'nykh moduley" [A Method for Testing Computation Modules]. In book: "Sintez, testirovaniye, verifikatsiya i otladka programm" [Program Synthesis, Testing, Verification and Debugging]. Riga: Zinatne, 1981, p 90-91.
12. "Otladka upravlyayushchikh algoritmov TsVM real'nogo vremeni" [Debugging of Real-Time Digital Computer Control Algorithms]. Moscow: Sov. radio, 1974, 328 pp.
13. Parkhomenko, P. P. and Pravil'shchikov, P. A. "Diagnostirovaniye programmnogo obespecheniya (obzor)" [Software Diagnostics (survey)]. AVTOMATIKA I TELEMEKHANIKA, 1980, No 1, p 117-141.
14. Pozin, B. A. "Metod strukturnogo postroyeniya testov dlya otladki upravlyayushchikh programm" [Structured Test Design Method for Debugging Control Programs]. PROGRAMMIROVANIYE, 1980, No 1, p 62-69.
15. Pravil'shchikov, P. A. and Shepin, V. S. "Sostavleniye strukturnykh programm v dialogovom rezhime s odnovremennoy generatsiyey testov" [Developing Structured Programs in an Interactive Mode with Simultaneous Generation of Tests]. AVTOMATIKA I TELEMEKHANIKA, 1979, No 8, p 129-138.
16. Syromyatnikov, V. P. "Modelirovaniye i otsenka nadezhnosti funktsionirovaniya programmnogo obespecheniya" [Modeling and Evaluating Software Operational Reliability]. In book: "Sovremennyye metody i sredstva programmirovaniya: Materialy seminara" [Modern Programming Means and Methods: Seminar Materials]. Moscow: 1981, p 122-128.

17. Toropov, N. R. "Kompleksnaya otladka programm na yazyke LYaPAS" [Complete Debugging of LYaPAS-Language Programs]. USiM, 1981, No 2, p 97-101.
18. Myers, G. J. and Hocker, D. G. The Use of Software Simulators in the Testing and Debugging of Microprogram Logic. IEEE TRANS. COMPUT., 1981, vc-30, No 7, p 519-623.
19. Schmitz, P., Megen R. and Bons, H. Methods of Systematic Test Case Determination and Test Case Preparation. Pract. Software Adapt. and Maint. Proc. SAM Workshop. Berlin, 1979; Amsterdam e.a., 1980, p 209-221.
20. Bergson, A. and Raud, R. "Ispol'zovaniye funktsional'no ekvivalentnykh moduley pri razrabotke programm dlya UTsVM" [Use of Functionally Equivalent Modules in Developing Programs for Digital Control Computers]. PROGRAMMIROVANIYE, 1981, No 4, p 50-56.
21. Kuryama, A. and Raud, R. "Informatsionnye modeli sredy dlya testirovaniya programm mikroEVM v SERP" [Informational Models of the Environment for Testing SERP Microcomputer Programs]. In book: "Sintez, testirovaniye, verifikatsiya i otladka programm" [Program Synthesis, Testing, Verification and Debugging]. Riga: Zinatne, 1981, p 138-139.
22. Raud, R. K. and Tamm B. G. "Sostoyaniye v oblasti programmirovaniya dlya mikroEVM (obzor)" [The State of the Art in Microcomputer Programming (Survey)]. PROGRAMMIROVANIYE, 1982, No 5, p 31-43.
23. Akulovskiy, V. G., Kudrin, V. G., Terent'yeva T. N. "Parametricheskaya sistema proyektirovaniya matematicheskogo obespecheniya sistem real'nogo vremeni" [Parametric System for Designing Real-Time System Software]. USiM, 1981, No 1, p 61-64.
24. Andrianov, V. A., Gringauz, A. D. and Bukin, A. Ye. "Povysheniye effektivnosti programmirovaniya zadach ASUTP mel'koseriynogo proizvodstva" [Increasing Task Programming Efficiency in an Automated System for the Control of Technological Processes Used in Small-Scale Production]. USiM, 1981, No 2, p 93-96.
25. Suleymanov, K. G. "Instrumentariy dlya modelirovaniya sistem real'nogo vremeni, realiziruyemykh na MVK 'El'brus'" [Toolkit for Modeling Real-Time Systems Implemented on "El'brus" Multiprocessor Computer Complexes]. PROGRAMMIROVANIYE, 1981, No 3, p 49-54.
26. "Sistema otladki programm obmena informatsiyey dlya spetsializirovannykh EVM" [Debugging System for Data Exchange Programs in Specialized Computers]. S. G. Goncharuk, V. I. Pustovarov, V. I. Salapatov and A. I. Flerov. USiM, 1981, No 6, p 70-72.
27. Iyudu, K. A., Bakhtizin, V. V. and Kasatkin, A. I. "Statisticheskoye modelirovaniye nadezhnosti i ustoychivosti programm" [Statistical Modeling of Program Reliability and Stability]. In book: "Sintez, testirovaniye, verifikatsiya i otladka programm" [Program Synthesis, Testing, Verification and Debugging]. Riga: Zinatne, 1981, p 109-110.

28. Messikh, I. G. and Shtrik, A. A. "Metodika i sredstva analiza struktury i kharakteristika slozhnykh kompleksov programm real'nogo vremeni. [Method and Means of Analyzing Complex Real-Time Program Packages]. In book: "Sintez, testirovaniye, verifikatsiya i otladka programm" [Program Synthesis, Testing, Verification and Debugging]. Riga: Zinatne, 1981, p 152-153.
29. Malinovskiy, B. N., Slobodyanyuk, A. I. and Pogorelyy, S. D. "Sistemnoye matematicheskoye obespecheniye mikroEVM i otladochnoy sistemy na baze mikroprotsessora K580IK80" [K580IK80 Microprocessor-Based System Software and Debugging System]. USiM, 1982, No 3, p 30-34.
30. Lipayev, V. V. "Konstruktivnyye pokazateli kachestva programm i svyaz' c tekhnologiyey ikh proyektirovaniya" [Structural Characteristics of Program Quality and the Role of the Design Technique in Program Quality]. IZV. AN SSSR. SER. TEKHN. KIBERNETIKA, 982, No 2, p 151-162.
31. Lipayev, V. V. "Analiz konstruktivnoy effektivnosti kompleksov programm real'nogo vremeni" [Analysis of the Structural Efficiency of Real-Time Program Packages]. USiM, 1982, No 5, p 47-54.
32. Boem, B., Braun Dzh. and Kaspar, Kh. "Kharakteristiki kachestva programmnogo obespecheniya" [Software Quality Characteristics]. Moscow: Mir, 1981, 208 pp.
33. Dolganyuk, I. N. and Karpovskiy, Ye. Ya. "Opredeleniye ekonomicheski optimal'nogo urovnya otlazhennosti programm" [Determination of the Optimum Economic Level of Program Debugging]. PRIBORI I SISTEMY UPRAVLENIYA, 1984, No 4, p 13-14.
34. Karpovskiy, Ye. Ya., Sagach, V. V. and Chernetskiy, A. A. "Nadezhnost' algoritmov upravleniya" [Control Algorithm Reliability]. Kiev, Tekhnika, 1983, 144 pp.
35. Goel, A. Z. and Soenjoto, J. Models for Hardware/Software System Operational-Performance Evaluation. IEEE TRANS. RELIAB., 1981, v 30, No 3, p 232-239.
36. Schmitz, P., Bons, H. and Megen, R. Methods and Techniques of Dynamic Program Testing. Pract. Software Adapt. and Maint. Proc. SAM Workshop. Berlin, 1979; Amsterdam e.a., 1980, p 193-208.
37. Turner, J. The Structure of Modular Programs. Commun. ACM, 1980, v 23, No 5, p 272-277.
38. Ayzenberg, Ya. Ye. and Konorev, B. M. "Organizatsiya imitatsionnogo modelirovaniya v avtomatizirovannykh sistemakh proizvodstva programm real'nogo vremeni" [Organization of Simulation Modeling in Automated Systems for the Production of Real-Time Programs]. USiM, 1982, No 4, p 83-85.

39. "Avtomatizirovannaya sistema proizvodstva programm SINTERM/Ya" [Automated System for the Production of SINTERM/Ya Programs]. Ya. Ye. Ayzenberg, I. V. Vel'bitskiy, B. M. Konorev, A. A. Stogniy. USiM, 1980, No 1, p 16-21.
40. Ayzenberg, Ya. Ye., Konorev, B. M. and Borzenko, A. G. "Tekhnologiya proizvodstva nadezhnykh programm" [Production Process for Reliable Programs]. In book: "Sintez, testirovaniye, verifikatsiya i otladka programm" [Program Synthesis, Testing, Verification and Debugging]. Riga: Zinatne, 1981, p 6-7.
41. Vel'bitskiy, I. V. and Sizov, A. A. "Sistema kompleksnoy otladki programmnogo obespecheniya ASUTP s ispol'zovaniyem dinamicheskoy modeli ob'ekta upravleniya (SKODINAMO)" [Complete Debugging System for Automated Technological Process Control System Software Through the Use of a Dynamic Model of the Control Objective (SKODINAMO)]. USiM, 1981, No 3, p 44-47.
42. Bichevskiy, Ya. Ya. and Borzov, Yu. V. "Razvitiye metodov simvolicheskogo testirovaniya programm EVM" [Development of Symbolic Testing Methods for Computer Programs]. AVTOMATIKA I TELEMEKHANIKA, 1982, No 8, p 93-101.
43. Bezborodov, Yu. M. "Individual'naya otladka programm" [Individual Debugging of Programs]. Moscow: Nauka, 1982, 192 pp.
44. Tai Kuo-Chung. Program Testing Complexity and Test Criteria. IEEE TRANS. SOFTWARE ENG., 1980, v 6, No 6, p 531-536.
45. Mashnikova, O. V. "Eksperimental'noye issledovaniye modeli nadezhnosti programmnogo obespecheniya pri avtomatizatsii ucheta zatrata na ego proizvodstvo" [Experimental Research on a Software Reliability Model in the Automation of Software Production Expenditure Estimating]. NAUCH. TR./MOSK. FIN. IN-T, 1981, p 12-17.
46. Duran, I. W. and Wiorkwoski, I. I. Quantifying Software Validity by Sampling. IEEE TRANS. OF RELIAB., 1980, v 29, No 2, p 141-144.
47. Iyudu, K. A., Kasatkin, A. I. and Bakhtizin V. V. "Prognozirovaniye nadezhnosti programm na rannikh etapakh razrabotki" [Predicting Program Reliability During Early Stages of Development]. NADEZHnost' I KONTROL' KACHESTVA, 1982, No 5, p 3-10.
48. Gmeiner, L. and Voges, U. Methods, Criteria and Automatic Tools for Software Testing. Pract. Software Adapt. and Maint. Proc. SAM Workshop. Berlin, 1979; Amsterdam e.a., 1980, p 183-192.
49. Musa, J. D. Validity of Execution-Time Theory of Software Reliability. IEEE TRANS OF RELIAB., 1979, v R-28, No 3, p 213-220.
50. Gordiyenko, A. V. "Testirovaniye pri otsenke dinamicheskoy korrektnosti programm ASU" [Testing in the Evaluation of Automated Control System Program Accuracy]. PROGRAMMIROVANIYE, 1982, No 6, p 48-52.

51. Kapyrin, V. A. and Khaletskiy, A. K. "Sposob otsenki nadezhnosti funktsionirovaniya programm real'nogo vremeni" [A Means of Evaluating Real-Time Program Operational Reliability]. PROGRAMMIROVANIYE, 1982, No 3, p 73-79.
52. Mohanty, S. N. Models and Measurement for Quality Assessment of Software. COMPUT. SURV., 1979, v 11, No 3, p 251-275.
53. Ramamoorthy, C. V., Ho, F. and Chen, W. T. On the Automated Generation of Program Test Data. IEEE ON SOFTWARE ENG., 1976, v SE-2, No 4, p 293-300.
54. Raykhlin, B. M. "Dialogovaya sistema otladki programm" [An Interactive System for Debugging Programs]. PROGRAMMIROVANIYE, 1982, No 2, p 84-89.
55. Gorskiy, V. G., Adler, Yu. P. and Talalay, A. M. "Planirovaniye promyshlennykh eksperimentov (modeli dinamiki)" [Planning of Industrial Experiments (Dynamics Models)]. Moscow: Metallurgiya, 1978, 112 pp.
56. Linger, R., Mills, Kh. and Uitt B. "Teoriya i Praktika Strukturnogo Programmirovaniya" [Theory and Practice of Structured Programming]. Moscow: Mir, 1982, 406 pp.

COPYRIGHT: Izdatel'stvo "Naukovo Dumka", "Upravlyayushchiye sistemy i mashiny", 1985

12746  
CSO: 1863/270

UDC 681.03

## DBS/R DATABASE CONTROL SYSTEM

Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 2, Mar-Apr 1985  
(manuscript received 31 May 84) pp 97-101

[Article by Yu. Sharr under the rubric: "Data Banks and Information Retrieval Systems"]

### [Text] Introduction

The "Robotron" combine (GDR) is producing YeS- and SM-series computers and developing a wide assortment of system and application software.

Work in the area of application software is proceeding along three basic lines:

1. Database system software, including data teleprocessing.
2. Application program packages to implement computerized mathematical methods.
3. Production resources for application program development.

The table below illustrates the basic product lines. Application program packages for the YeS computer series have received commercial approval in various types of automated systems in the GDR as well as in other countries using YeS computers.

The database software developed at the Robotron combine includes database management systems (DBMS) and information retrieval systems.

The BASTEY and SAW1 database management systems and the AIDOS information retrieval system, developed in the YeS DOS [disk operating system] environment, have found broad acceptance in the GDR and abroad. The experience gathered in producing and installing these systems and the scientific research carried out by the Robotron combine has permitted the development of a new generation of world-class database software running in the YeS OS [operating system] environment. This generation includes the DBS/R database management system and the AIDOS information retrieval system featuring teleprocessing support systems; ESY/R is used for DBS/R and DAKS for AIDOS. In keeping with modern concepts there is also the DABA/1600 DBMS for SM computers.

## Basic trends in application software development by the Robotron combine

### YeS computer series

Systems	YeS DOS	YeS OS in MFT, MVT and SVS modes	SM computer
DBMS	BASTEI SAWI	DBS/R	DABA/1600
Information retrieval	AIDOS	AIDOS	
Data teleprocessing		ESY/R, DAKS	
Mathematics	Discrete optimization Transport optimization Digital math Network planning OPSI, SIMDIS, STATISTIK	Digital mathematics Network planning OPSI-2, SIMDIS-2, STATISTIK-2, DISKO-2 EDO-2, DISKO-T	MALEQ-1600 TRANSPORT-1600 MASTAT-1600
Engineering systems		TESYS	SEP-1600

Let us examine one of these systems, the DBS/R database management system.

#### DBS/R general features

This system includes all programming resources needed to create and maintain databases with an unlimited number of data files and without restrictions on relationships between these files. It has language resources available for retrieving any data in these files.

DBS/R supports a network data model based on CODASYL recommendations with some modifications allowing the representation of list, tree and network structures of any size and complexity and supporting traditional file organization methods. Thus, it provides the user with a single system for handling the most varied of data organization structures and methods.

The DBS/R operates in the YeS OS environment (Version 4.1 and up) on all YeS computer models. The following minimum hardware configuration is required:

- At least 512K-bytes of main memory
- One operator console
- One input device with card punch
- Two to four disk drives
- Two magnetic tape units
- One alphanumeric printer

The DBS/R DBMS also runs on IBM/360 and /370 series computers.

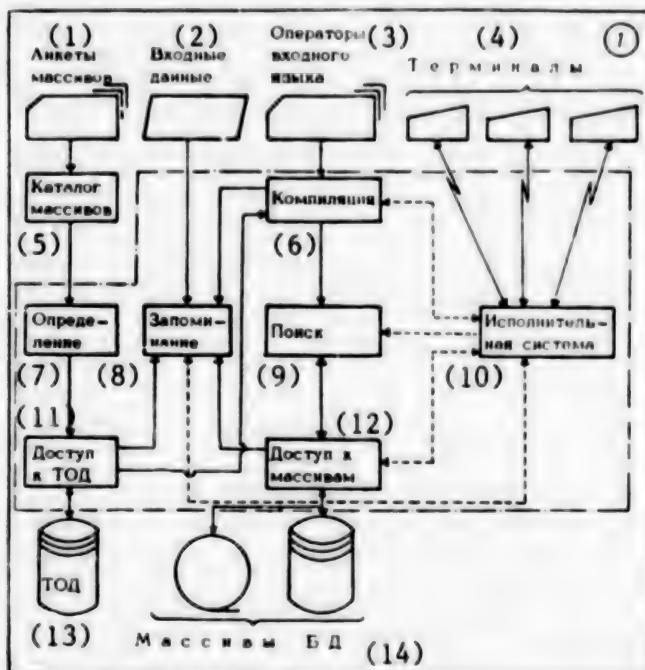


Figure 1. Overall block diagram of the DBS/R database management system

1--File forms	8--Storage
2--Input data	9--Search
3--Input language operators	10--Execution system
4--Terminals	11--Access to data description table
5--File catalog	12--Access to files
6--Compilation	13--Data description table
7--Definition	14--Database files

#### DBS/R functions and languages

The following basic functions are provided:

- Definition, i.e. describe file structure and relationships and describe field processing operations.
- Storage, i.e. create, maintain (update) reorganize and restructure data files.
- Search, i.e. search for any information in the database according to given criteria and process, prepare and output this information.

All these basic functions are in turn supported by functions which assure data security (by preventing unauthorized access) and the physical and informational integrity of files.

Figure 1 shows a block diagram of the DBS/R database management system.

System users have two language resources available for controlling DBS/R operations: a data description language and a data manipulation language.

The data description language provides a means of defining the data structure (describing file characteristics and their record and relationship structure), as well as a means of assigning operations to assure the informational integrity of the data. It allows the description of conditions which a given field must satisfy when entered in the database and of monitoring or recoding operations whose execution is required on data entry or output. An example of possible processing operations during data entry is illustrated in Figure 2.

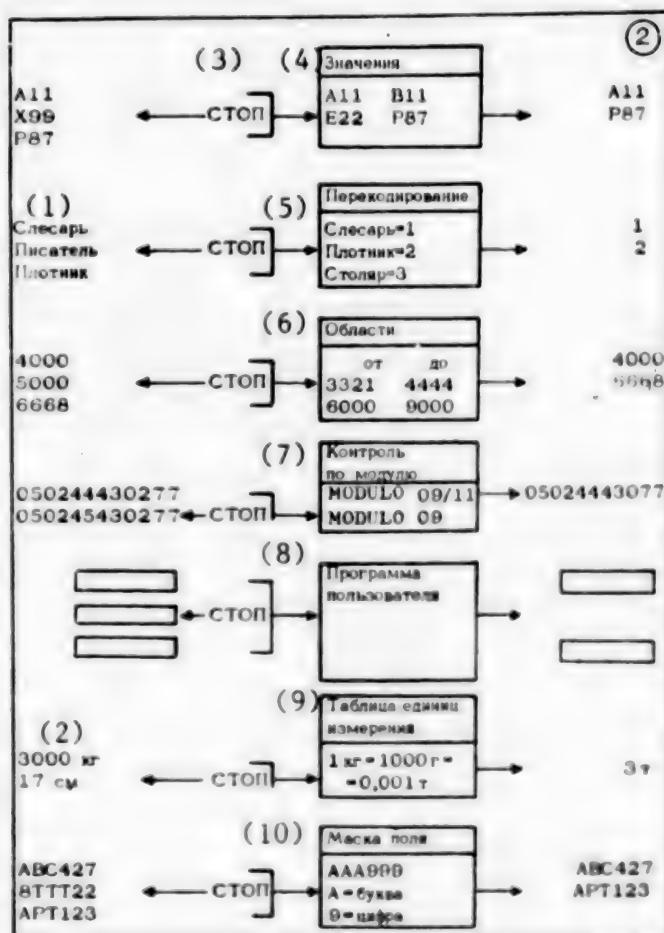


Figure 2. Examples of entry data processing operations

1--Fitter  
    Writer  
    Joiner  
2--kg  
    cm  
3--Stop  
4--Values  
5--Recoding  
    Fitter = 1  
    Carpenter = 2  
    Joiner = 3

6--Areas  
    from      to  
7--Modulus check  
8--User program  
9--Measurement unit table  
    1 kg = 1000 g = 0.001 t  
10--Field mask  
    A = character  
    9 = digit

The data description with its data integrity assurance operations are stored in the data description table.

The data security rule description (access privilege statement) facility is a special subset of the data description language which protects any data element (files, record type, segment or field) from unauthorized access at the data type level (for example, allowing access only to certain record fields) as well as at the level of specific record types, when access is allowed only upon satisfaction of certain conditions which depend on data content. The access privilege is assigned separately for each DBS/R function.

The storage, search and physical integrity functions are controlled through DBS/R data manipulation language commands. This is a high-performance procedural language which can be used alone or in connection with the database. The same data manipulation language commands are used in both applications. Assembler, PL-1 and COBOL can be used as database languages. The DBS/R data manipulation language is based on syntactical structures close to those of a restricted natural language, thereby facilitating learning and use.

Experience shows that non-programmer users can create simple programs after a three-day training period in this language; two weeks of training is sufficient for rather complex programs.

At the present time there are German- and English-language versions of the DBS/R data manipulation language.

The DBS/R data manipulation language contains functions which carry out elementary and group (combined) functions, such as access to individual database records or to sets of records, sequential file input/output, output list formatting and simple processing functions (comparison, arithmetic operations, etc.), as well as specialized discrete functions intended for estimate automation in engineering preparation for production.

A portion of a product structure breakdown program is shown below and its result is illustrated in Figure 3.

```
RETRIEVE EXECUTE;
DECLARE RLEVEL 8;
DECLARE IAREA RECORD 13;
DECLARE KEY 13;
INPUT TO IAREA;
READ FROM MMPART BY A KEY;
MOVE FROM MMPART PARTNO 'RPARTNO' FROM PARTNAME 'RPARTNAM';
OUTPUT;
HEADING 'SPETSIFIKATSIYA' RPARTNO RPARTNAM;
FEED;
HEADING 'UROVEN' RAZUZL. NO. IZD. NAZVANIYE IZDELIYA # YED. IZM.'5;
FEED;
HEADING;
STRUCTURE VIA MMPART IN MCSTRU WITH STRUCLEVEL EXTERNAL TO RLEVEL ON STRUCEX2
```

```
OUTPUT;  
OUTPUT OUTPUT;  
LINE RLEVEL 5 MMPART PARTNO 15 PARTNAME 30 UM 65;
```

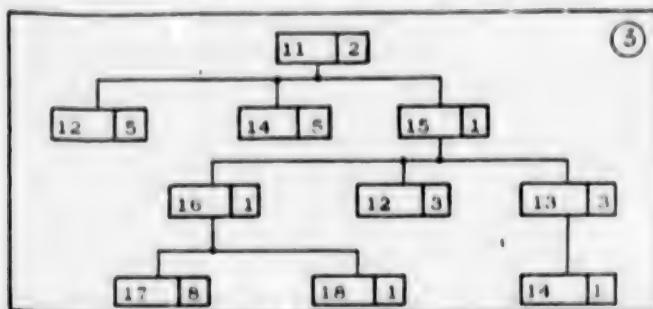


Figure 3. Product structure produced by the structure breakdown program

The breakdown program is an example of independent use of the DBS/R data manipulation language.

Data manipulation language commands supporting the storage function (entry, updating, etc.) are highly complex. For example, one command can control the processing of an entire input file whose individual records can in turn initiate changes in all database files. During this process all data integrity operations contained in the data description table are also executed.

The physical integrity of data is also assured by data manipulation language commands which protect the database and intermediate program operating results from disturbances caused by hardware or software errors as well as from operator error. Any database condition is restorable using the physical integrity language resources. The DBS/R also contains a function used to change the database structure in order to increase program operating efficiency or to develop applications.

The DBS/R data description and data manipulation languages provide a high level of data autonomy for programs written using the data manipulation language.

#### Modes and types of DBS/R operation

The system in question provides users two methods of operation: standard processing and real-time processing.

In the standard method of operation previously compiled data manipulation language programs are stored in a library of loading modules from which they can be called and used repeatedly without recompilation. This type of operation is designed for the execution of previously planned tasks (queries).

Real-time processing is designed for the execution of operational tasks not planned in advance. In this application, the DBS/R data manipulation language is used in a stand-alone form and programs are executed immediately after compilation.

Both methods can operate in either the batch or the teleprocessing modes.

#### Comparison of DBS/R with other database management systems

**Storage structure:** The DBS/R system can create an exceptionally flexible storage structure and includes most of the features found in other DBMS. The capability of varying database structure to increase system efficiency is significantly broader with DBS/R than with other systems.

**Database restoration and input data processing:** In this area DBS/R stands out favorably from other CODASYL-model database management systems. These DBMS do not support input data processing or record analysis and synthesis,<sup>3</sup> leaving these functions to be performed by application programs. In the DBS/R system, data manipulation language commands and storage functions automatically initiate (according to instructions contained in the data description table) all actions required for record analysis and synthesis, as well those needed for record field processing during input.

**Completeness of restoration functions:** All the database management systems compared allow the creation or alteration of a single database record (by a single data manipulation language command). In the DBS/R system one input record causes changes in several database files, providing an even higher level of consistency. Other systems require several data manipulation language commands to perform the same tasks, thus all efforts to ensure data logic consistency are handled by application programs.

**Completeness of search functions:** In the DBS/R database management system these functions are supported by data manipulation language commands which allow parallel reading of various types of interrelated records. With other systems even this function requires several data manipulation commands, each one of which accesses only one type of record.

**DBS/R data manipulation language user-friendliness:** Just as the ADABAS system query language, the DBS/R data manipulation language is similar to a natural language. It supports combined status checking functions, simple arithmetic functions, report generation and outputting and data display on video terminals. With these features, DBS/R is comparable to the ADABAS system but it has more capabilities than the IMS and CODASYL systems.

As with systems supporting the CODASYL model, DBS/R provides the capability of directly including data manipulation language commands in the host language (assembler, COBOL, PL-1). In this sense the IMS system is inferior to DBS/R because IMS access to the database takes place via assembler macroinstructions and the link with other languages is implemented at the intermodule connection level.

**Production data processing system orientation:** DBS/R's storage structure properties and some of its specialized data manipulation language commands are oriented toward a rather broad and important field of application--production systems--especially in the area of engineering preparation for production (a set of data manipulation language commands for structure breakdowns and applicability evaluation, as well as convenient means for generating required

engineering and design documentation. CODASYL-model based systems, and the ADABAS and IMS systems, require considerably greater expenditures on application programming for these applications.

**High security during DBS/R operation:** The DBS/R database management system assures the consistency and integrity of database files, data descriptions and programs. Other systems require special measures implemented by means of application programs. Data protection from unauthorized access is provided at the data type level and at the level of specific data examples. Access privileges can be defined separately for any of the DBS/R functions.

#### DBS/R's range of application

This system has been available to users since 1979. At present more than 200 users in the GDR and over 20 foreign users have acquired DBS/R systems. A large number of these users purchased the system for multiple-user applications in factories in all branches of industry. The numerous DBS/R applications include traditional fields as well as those involving DBS/R's in the creation of highly integrated data processing systems—even complete automated management systems.

The DBS/R system finds its broadest application in the plants and combines operated by the following ministries:

Ministry	Number of plants
Electronics and electrical engineering .....	41
Heavy machine construction .....	14
Machine tool industry .....	27
Mining and metallurgy .....	15
General, agricultural and transport machine construction .....	9
Chemical industry .....	11
Light industry .....	8
Construction .....	4

Moreover, the DBS/R database management system has been introduced in the following areas: agriculture, forestry, the food industry, postal and telegraph operations, health care, higher and middle specialized educational facilities, geology, city cultural centers, etc.

Nearly all users have their own computer center and install the DBS/R independently on the basis of Robotron combine standards consultations. A large number of small enterprises share the DBS/R system through regional computer centers.

The DBS/R database management system finds its widest application (nearly 40%) in the area of engineering preparation for production. Let us examine a brief picture of basic applications.

**Engineering preparation for production:** The heart of the database consists of files containing design and engineering data on all components and

assemblies produced. These data are used to prepare design and engineering specifications, bills of materials, work station utilization lists, process steps and many other documents.

**Personnel database:** The DBS/R system is used to update the database when changes occur in wages, occupations, classifications, etc. Standard tasks are performed periodically: monthly wage estimates, service bonuses for workers (completion of monetary transfers, housing payments, etc.).

**Management information system:** These systems are developed for plant, combine and ministry management. They are used for planning, balancing accounts, preparing plan estimates and monitoring schedules and decisions.

#### Economic effect of DBS/R introduction

In Ministry of Electronics and Electrical Engineering plants and combines the DBS/R management system is used primarily to improve engineering preparation for production, especially to solve the following problems:

- Automated production of highly integrated microcircuits
- Automated design of instrumentation and devices
- Production and maintenance of engineering documentation
- Improvement of auxiliary and standard processes involved in scientific and engineering preparation for production
- Creation and maintenance of plant engineering and process data banks

Fifty seven groups of tasks based on the DBS/R system have been introduced at this ministry's 13 combines (41 plants). The following economic effect indicators have resulted:

- A significant decrease in the time required to produce design and engineering documentation allowed an average savings at each plant of 12 thousand working hours for production engineers.
- The release of 350 production engineers at all ministry plants due to improvements in auxiliary and standard processes.
- Significant savings in expenditures for data processing system creation and development, which allowed savings of 80 percent in comparison to the use of ordinary programming languages.

Unanticipated queries can be programmed and executed within minutes.

Further development of the DBS/R system continues at this time, especially in the area of increasing program operating efficiency (program execution times) and expanding physical data integrity assurance capabilities.

COPYRIGHT: Izdatel'stvo "Naukovo Dumka", "Upravlyayushchiye sistemy i mashiny", 1985

12746  
CSO: 1863/270

UDC: 025.4.036:681.3.06

SOME PROBLEMS IN THE TECHNOLOGY OF CREATION AND INTRODUCTION OF DATABASES  
ON MACHINE-READABLE MEDIA

Moscow KLASSIFIKATORY I DOKUMENTY in Russian No 12, Dec 84 pp 22-29

SERGAZIN, Zh. F., Doctor of Technical Sciences, Moscow State Historical  
Archives Institute and TOLOCHKO, V. V., All-Union Scientific Research  
Institute of Documentation and Archive Matters

[Abstract] A study is made of the process of improving data security in a database by frequent backup of data from magnetic disk to magnetic tape in several generations. Time studies indicate the relationship between the frequency and duration of backup operations and the amount of useful machine time during which the database is actually available for use. Since archival backup requires restoration of parts of the database from several generations of backup in the event of a system disaster, time saved in the backup cycle is expended in the restoration cycle following data loss. Equations are derived which define computer utilization factors for various backup strategies. Figures 3; reference 1 Russian.

[140-6508]

UDC: 681.3

FORMALIZED STATEMENT OF PROBLEMS OF SYSTEMS DESIGN OF TERMINAL SYSTEM  
SOFTWARE

Riga AVTOMATIKA I VYCHISLITEL'NAYA TEKHNIKA in Russian No 4, Jul-Aug 84  
(manuscript received 10 Jan 84) pp 10-17

PIROGOV, V. V. and BATRAK, V. N.

[Abstract] The use of microcomputers for the creation of terminal systems has resulted in the appearance of qualitatively new characteristics requiring restudy of architectural decisions and organization of design automation systems. This article presents a systems level program structure for terminal systems. The use of microcomputers allows terminal systems to be

implemented in various architectural versions: single-processor architecture, including local area networks. The systems model of terminal system software combines object and process models. Methods of its decomposition are analyzed, as are conditions of the transition to logic models in the stage of logical design. The terminal system software is presented as a system of sequential processes interacting by exchanging messages. The solution allows for further development in systems without shared memory. The concepts of abstract data types and parallelism are central to the methodology. References 13: 6 Russian, 7 Western.  
[137-6508]

UDC: 681.3

#### MODELING OF ASSOCIATIVE MEMORY FOR RELATIONAL DATABASES

Moscow PROGRAMMIROVANIYE in Russian No 6, Nov-Dec 84  
(manuscript received 12 Feb 82; revised version 20 Jan 84) pp 45-51

FROLOV, O. R.

[Abstract] The major principle of organization of memory structure utilized in modern DBMS is the system of multilevel indexing. Control of multilevel files requires that the system create three to four bytes of system information for each byte of information input. In this work, while preserving this redundancy, it is suggested that memory structure be organized slightly differently so as to preserve the possibility of constructing relational databases with ordinary DBMS facilities. Data are stored in so-called K structures, maintaining ordered storage of the values of an attribute by use of multilevel indexed references. Data manipulation algorithms are described. Notes are made on how K-structures can be used with the Bank, SEDAN, and OKA DBMSS. Figures 4; references 3 Russian.  
[153-6508]

UDC: 681.3.06

#### SOME PROBLEMS OF THEORY OF RELATIONAL DATA PROCESSING LANGUAGES

Kiev KIBERNETIKA in Russian No 5, Aug-Sep 84  
(manuscript received 20 May 81) pp 38-42

VERNIK, L. V. and VINITSKIY, I. M., Scientific Research Institute of Psychology of the UkrSSR Academy of Sciences

[Abstract] An expanded relational algebra (ERA) and semantically equivalent relational calculus (beta calculus) are presented, and an algorithm for conversion of an arbitrary beta expression into the semantically equivalent

ERA algorithm is described. The ERA and beta calculus are relational data processing languages. Since they require significant mathematical education to be understood, the languages are not suitable for nonprofessional users. However, they can serve as the mathematical foundation for the construction of user-friendly procedural and declarative data processing languages. Examples are RYAOD and DNEPR, data processing languages with expanded sets of functional and service capabilities. References 10: 6 Russian, 4 Western.  
[204-6508]

UDC: 681.327.01

#### MESSAGE OUTPUT CONTROL IN THE INES SYSTEM

Moscow PROGRAMMIROVANIYE in Russian No 6, Nov-Dec 84  
(manuscript received 4 Jul 83) pp 52-57

GODUNOV, A. N., YEMEL'YANOV, N. Ye., ROMANOV, A. P. and SVERDLOV, S. S.

[Abstract] A message output system is included in the INES DBMS. The hierarchy of output facilities can output anything from a complex document to a single row of text to the display screen, printer or disk file. Accessing the display facility from Assembler, FORTRAN, PL/I and INES languages is described. The ULINE facility of INES instead of the OPEN and PUT commands of the YeS operating system provides simpler programming, expanded functionality and superior control; for interactive terminal use with OS YeS, more than 1 MByte main memory is required. References 3 Russian.  
[153-6508]

UDC: 681.3.06

#### COMPLEXITY OF STRUCTURAL TESTING OF PROGRAM MODULES WITH LOOPS

Moscow PROGRAMMIROVANIYE in Russian No 6, Nov-Dec 84  
(manuscript received 22 Dec 83) pp 70-77

LIPAYEV, V. V. and POZIN, B. A.

[Abstract] Full testing of every possible path through a program with several loops is extremely complex. A criterion on structural complexity of programs is introduced to allow estimation of the complexity of structural testing of logical programs with loops. Equations are derived for computation of the complexity of testing as a function of the area covered by a graph of a loop. The relative estimate of the complexity of a portion of a program structure enclosed by a loop and the influence of this portion on the structural complexity of the entire program module allows an approximate estimate to be generated for the complexity of testing of several types of

modules with loops. Exhaustive testing is so complex even in programs with relatively few loops that it can be used only in exceptional cases.  
Figures 4; references 6: 5 Russian, 1 Western.  
[153-6508]

UDC: 681.3.06

PRINCIPLES OF DESIGN OF A TOOLKIT FOR DEVELOPMENT OF SOFTWARE FOR AUTOMATION AND CONTROL SYSTEMS BASED ON SM COMPUTERS

Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan 85  
(...manuscript received 10 Jan 84; after revision 6 Apr 84) pp 47-52

KIBITKIN, V. V., Leningrad Institute of Analytical Instrument Building of the USSR Academy of Sciences

[Abstract] The key to development of software for systems of automation and control (SAU) is the drawing of a deep analogy between its development and the process of production of an ordinary industrial product. Development of SAU software therefore requires computer hardware, systems software, documents determining the composition and sequence of operations to be performed by the software and the necessary systems software for each operation which the end product must perform. All these requirements together are described as a toolkit for the development of SAU software. The main requirements for the toolkit are that it allow development of software by many users in a time-sharing mode, that it support storage and utilization of common libraries of routines and common databases, and that it allow each user to interact with the toolkit during development of the software. A toolkit of this type for use in development of SAU software for the SM-4, SM-3 and 'Elektronika-60' computers is described. PASCAL is recommended as the high-level development language, while the system language should be PL/11, based on MACRO-11. One and two-machine configurations and various systems, such as bigTOS, multiTOS, microTOS, polyTOS, OS RV, RAFOS, are mentioned. An alternative to what the author built would be a system based on R-technology, which has been proven to have serious advantages on YeS and BESM-6 machines, but has received little use so far on SM machines.

References 4 Russian.

[196-6508]

UDC: 681.3.068

## GRAPHICS SOFTWARE SYSTEM FOR MACHINE GRAPHICS SYSTEM

Moscow PROGRAMMIROVANIYE in Russian No 6, Nov-Dec 84  
(manuscript received 19 Dec 83) pp 78-82

POZDNEYEV, S. A.

[Abstract] Graphics software must be sufficiently universal to allow its utilization in a variety of situations for the production of a number of different types of graphics constructions on various output devices. At the same time, it should be relatively easy to use, allowing the user to concentrate on the graph being produced rather than the intricacies of the software. This article presents an attempt to satisfy these requirements, suggesting a system of graphics subroutines allowing a graphics dialogue to be organized with a user program so that the user can develop mobile machine graphics systems in FORTRAN. The primary advantage of the software suggested is the presence of universal graphics elements and functions, each implementing an individual graphics user requirement: construction of defined tabular functions, rectangular grids, histograms, equilateral polygons, various arrows, various types of lines, connection of points into a smooth line, connection of points by lines with center characters, construction of spirals, ellipses, circles, plotting of curves defined as polynomials and construction of solid images. The programs have the advantages of mobility, simplicity of use and virtual independence of type of graphics output device. The package of programs was used in the UMPLOT system on a CDC CYBER-172. References 24 Russian.

[153-6508]

UDC: 681.3.06

## IMPLEMENTATION OF THE GRASS GRAPHICS SOFTWARE IN THE SM COMPUTER ENVIRONMENT

Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan 85  
(manuscript received 16 Jul 84; after revision 20 Jul 84) pp 112-113

KURILOV, M. A.

[Abstract] A study is made of some problems of implementing graphics software on an SM-3 computer using the RAFOS operating system. The switch from the YeS to the SM-3 environment was conducted so as to preserve full input language compatibility for application programs. An expanded set of high level graphic primitive functions was included in SM GRAS, supporting construction of arcs of circles and other second-order curves, curve matching, matching of straight lines or arcs with each other, shading of various areas and construction of portable elements, roughness characteristics, dimension lines and explanatory text. References 5: 4 Russian, 1 Western.  
[196-6508]

UDC: 681.306

PRESENTATION OF GRAPHIC INFORMATION ON RIN-609 DISPLAY SCREEN

Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan 85  
(manuscript received 6 Jun 83) pp 122-123

AVETISOV, L. G., VASINYUK, I. Ye. and NANASYAN, A. S.

[Abstract] In a standard RIN-609 display system the screen information is regenerated once per power supply cycle. In practice, regeneration utilizes only 25% of available machine time. In the device here described, this time is used to present graphic information stored in display RAM on the screen. Thus, alpha-numeric and graphic information are simultaneously and independently displayed on the screen. The graphic information can be presented in either point or vector mode. Graphs are constructed by successive switching on of points and vectors stored in RAM by means of digital-analog converters allowing, in contrast to raster presentation, construction of graphs of significant complexity with limited RAM. Figure 1.  
[196-6508]

UDC: 681.3

ASSEMBLER 2 PROGRAMMING SYSTEM FOR YeS OS

Moscow PROGRAMMIROVANIYE in Russian No 6, Nov-Dec 84  
(manuscript received 12 Aug 83) pp 82-87

VOYUSH, V. I., DEGTYAREVA, G. S., BELIKOVA, G. I., DASHKEVICH, V. M. and PARAKHNEVICH, N. N.

[Abstract] Assembler 2 for the YeS computers under OS is the latest version of Assembler for the 'kyae-2' YeS computers using version 6.1 of the operating system. It improves upon Assembler of OS 4.1, the final OS version for YeS-I machines, and upon Assembler of OS 6.1, for YeS-II machines. The major specific feature of Assembler 2 is a significant increase in the assembly speed due to the use of new algorithms which utilize larger areas of RAM. Assembly speed has been increased by a factor of 2-3 over Assembler, increasing to as much as a factor of 5-6 for certain types of programs. Assembler 2 also preserves upward compatibility, expands the language, provides improved diagnostics, allows debugging of macros, allows batch processing and adjustment of Assembler parameters. Assembler 2 uses 2 passes as opposed to Assembler 6.1, which used 3, and previous version, which used 4. The new capabilities of the language are briefly described along with the peculiarities of implementation of Assembler 2 on YeS computers running OS. Assembler 2 can be used in all versions of YeS OS beginning with 4.1, as well as the time-sharing system and the dialogue processing subsystem of the virtual machine system (SVM) of YeS computers. The minimum

requirement is 180 KByte RAM in MFT or MVT, 192 KByte in SVS. The distribution tape includes various means for inserting Assembler 2 in OS Yes or SVM Yes, including the ability for user-defined parameters. Figures 2; references 4 Russian.

[153-6508]

UDC: 681.3.51

#### TRANSLATION OF APL SYSTEM STATEMENTS FOR THE SM-1 COMPUTER

Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan 85  
(manuscript received 26 Apr 83; after revision 4 Aug 83) pp 65-70

KONDRAZHEV, A. V.

[Abstract] The specifics of APL which prevent the production of an APL compiler are noted. However, a precompiler has been developed which typically compiles over 90% of the text of ordinary programs. This article discusses a further development of this method for use in the APL/SM-1 dialogue system developed by the author. The use of the precompilation method allows the construction of an optimizing APL interpreter, completely impossible when direct full interpretation is used. References 5:  
3 Russian, 2 Western.

[196-6508]

UDC: 681.154

#### DEVELOPMENT OF THE FORT-P PROBLEM-ORIENTED LANGUAGE

Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan 85  
(manuscript received 30 Mar 83; after revision 13 Dec 83) pp 77-80

SMOL'NIKOV, V. A.

[Abstract] Several problems are discussed related to the development and implementation of a high-level language for a problem-oriented computer system designed to perform a broad range of scientific and engineering tasks which can be reduced to repeated solution of systems of algebraic, integral and differential equations. The computer system is a multiprocessor MIMD system with functional separation of data storage, primary and secondary operations. The architecture allows pipeline organization of computation on two levels. There is a broad set of nonstandard I/O devices, and the software must include a broad range of application programs and programming language processors. The problem-oriented language FORT-P is based on FORTRAN-IV to allow use of the library of programs already available. It includes two new data types: CONTROL and FIXED, the former used to synchronize

parallel processes. Data control in the language is static. The language includes facilities for writing programs at the conceptual level for a given object area, including facilities for parallel and pipeline processing of information as well as macroprocessing. Flexible resource control is implemented. An optimizing compiler is provided which compiles more slowly but produces more effective object code, close to manually optimized code. FORT-P supports the use of structured programming and debugging. A Yes FORT-P compiler consists of 5,000 PL/I and assembler commands, and took about six man-months to develop. References 7: 6 Russian, 1 Western.  
[196-6508]

UDC: 681.3.06

#### SORT-7/SM SORTING SUBSYSTEM

Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan 85  
(manuscript received 29 Dec 83) pp 70-73

KVACHUK, K. P.

[Abstract] The SORT-7/SM software subsystem has been developed to meet traditional sorting needs for the SM-3, SM-4 computer family. The subsystem is designed to sort data files under the control of the KV operating system. SORT-7/SM provides the user with the capability to organize the process of sorting an input file in accordance with the requirements and limitations of the specific application and is thus a universal sorting subsystem. The input file may contain fixed or variable length records. The maximum number and length records is limited by the computer hardware. The position and length of keys within records, sorting rule for each key and order of keys are input by the user. Binary or alphabetical sorting in increasing or decreasing order may be used for each key. SORT-7/SM has been tested and was transmitted in 1983 to the State Fund of Algorithms and Programs. It is now distributed by the Republic Fund of Algorithms and Programs, Institute of Mathematics, Belorussian Academy of Sciences. It arose as part of the development of ASPID-7. Figure 1; references 7 Russian.  
[196-6508]

UDC: 681.3.06

#### DATA MODELS IN DIALOGUE INTEGRATED AUTOMATIC PLANNING SYSTEM FOR INDUSTRIAL ENTITIES

Kiev KIBERNETIKA in Russian No 5, Aug-Sep 84  
(manuscript received 11 Apr 83) pp 43-47

DODONOV, S. B. and VISIKIRSKIY, V. A., Institute of Cybernetics, UkrSSR Academy of Sciences

[Abstract] The graphic models most widely used in design of the data base management system, applications programs and basic operations with data in

automated industrial design systems are studied. Conflicting demands placed on these models include complete and adequate description of the object being designed with simultaneous effective implementation independent of specific application programs. This has required a three-level organization of data, with external schemata (graphic models), each of which defines the object area of the user, at the first level; the conceptual or generalized model of the associative structure, used to reflect the set of external models, at the second level; and the internal model, defining the placement of data on physical information storage devices, at the third level. This allows construction of modular subsystems or application programs which interact with and control all aspects of design and preparation of production, the use of various levels of data abstraction, allowing transition from one model to another, and simplification of the exchange of models and interaction of various users such as engineers, designers and technicians. In the approach suggested in this article, external models are divided into six basic types which are described by graphs, the nodes and edges of which may correspond to data or operations, while the 'horizontal' connections are assigned by associative relationships. The basic operations of processing models described by nonweighted graphs of these types have been implemented on an SM-4 computer. Figures 4; references 8: 7 Russian, 1 Western.

[204-6508]

UDC: 681.3.06

#### GAMES MODELING OF DIALOGUE IN INTERACTIVE SYSTEMS

Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan 85  
(manuscript received 28 Jun 83; after revision 25 May 84) pp 73-76

YUSHKOV, S. A.

[Abstract] The major problem in developing a dialogue between man and machine is that of developing a model of the dialogue allowing organization of effective man-machine interaction in performing tasks which cannot be completely formalized. Construction of a dialogue based on a network model allows correlation of services provided by the computer with definite points in the dialogue, built into the algorithm as it is developed. As a result, as the capabilities of systems increase the complexity of the graph structure increases, making its implementation more difficult. Network modeling little uses the major advantages of dialogue interaction, including the thinking capacities of the partners to make the process more dynamic. The degree of activity of the man-machine dialogue can be increased by using the principle of games theory to model the dialogue in interactive systems. In this case the dialogue can be considered a game of two layers with non-contradictory goals. This type of modeling was used on an 'Elektronika T3-29' minicomputer to develop an information retrieval system, the dialogue imitated was two matrices; one for the computer and one for the user. The use of games theory to model a dialogue allows a significant increase in the level of activity of the computer. References 13 Russian.

[196-6508]

DESIGN OF THE BASE LANGUAGE FOR AN INFORMATION MACHINE

Moscow DOKLADY AKADEMII NAUK SSSR in Russian Vol 280, No 2, Jan 85  
(manuscript received 4 Apr 84) pp 309-313

AYLAMAZYAN, A. K., GILULA, M. M., 'Poisk' Scientific-Production Association, Moscow

[Abstract] A study is made of a virtual information machine which, by using the text of an algorithm in a certain formalized language called the base language, performs a desired sequence of actions. The abstraction of the information machine allows the study of a number of problems which are important for the theory and practice of general purpose information systems. One is to find a model of data and methods of presentation of algorithms, i.e., to construct a base language, guaranteeing that the information machine can perform any algorithm written in that language for an arbitrary state of its finite memory. The base language is logically complete relative to a certain applied predicate calculus. This article presents a model of one class of logically complete base languages providing independence of algorithms from data. References 3: 1 Russian, 2 Western.

[175-6508]

UDC: 681.3.06

LANGUAGE IMPLEMENTATION OF PARALLEL ASYNCHRONOUS COMPUTATION MODEL

Kiev KIBERNETIKA in Russian No 5, Aug-Sep 84  
(manuscript received 24 Nov 82) pp 32-37

LEL'CHUK, T. I., Computer Center, Siberian Branch, USSR Academy of Sciences

[Abstract] A description is presented of an asynchronous parallel model of computation and its implementation in the POLYAR language, reflecting the asynchronous programming method. All program segments are considered initially parallel, independent and nonordered, and are interrelated in the language by assignment of limitations. This approach differs in principle from series-parallel organization, which simply indicates parallel segments and synchronization facilities in a basically sequential structure. The structure of the model described is hierarchical with asynchronous control of the computational process, combining common and distributed memory and parallel processing of structured data, with the accent given to flow control and distributed memory. The orientation of the language toward systems applications has determined a number of technological requirements such as programming reliability, tolerance and easy combination of programs plus expandability of the system. Some requirements are satisfied because the language is based on a conceptually unified model of computation containing a modular hierarchical organization and by the nonprocedural representation of control. Other specifics of the language include the necessity of static definition of all program objects, the possibility of assigning attributes to

variables influencing their representation, the presence of the apparatus of types for description of variables, separation of the description of computations from the context in which the computations occur and the parametrization of all descriptions, which can be made specific at various levels of passage of a program through the computer system. Figures 3; references 5: 4 Russian, 1 Western.

[204-6508]

UDC 518.5

#### COORDINATE TRANSFORMATION ALGORITHMS FOR MICROPROCESSORS

Leningrad PRIBOROSTROYENIYE in Russian No 4, May 85  
(manuscript received 11 Oct 84) pp 34-39

ANISHIN, N. S., Kuban State University

[Abstract] Processing of microprocessor operating or measuring information frequently requires algorithms for converting Descartes system coordinates into polar coordinates, and vice versa, as is done by the CORDIC system. Most systems are unable to do multiplication, division or square root operations without specially prepared programs. The present article offers a series of algorithms intended to transform Descartes coordinates into polar data and the reverse. The solutions are based on a digital circular interpolation. Both algorithms and the process followed to arrive at them are presented. The author finds that the relationship of coordinate to diagonal pitch is equal to  $\sqrt{2}$ , with every  $45^\circ$  of the arc having  $\sqrt{2}R/2$ , while the entire  $360^\circ$  has  $4R/2$ . By using whole numbers the program is simplified to where it requires only half the computer memory needed for previous algorithms. Various applications are suggested. Figures 2; references 3: 2 Russian, 1 Western.

[356-12131]

## APPLICATIONS

### SHIPBOARD COMPLEX PROCESSES INFORMATION FROM WEATHER SATELLITES

VODNYY TRANSPORT in Russian 25 Jun 85 p 3

[Text] A hardware-software complex developed at the Far East Research Center's Institute of Automation and Control Processes is called upon to aid scientists who are studying the ocean. This complex has automated the processing of information received on scientific research ships from weather satellites.

While intended chiefly for receiving satellite information, the complex performs other tasks, too. For example, it collects readings from many sensing devices deployed in the ocean, freeing specialists for more creative work. In addition to its purely scientific uses, the new system enables navigators quickly to update weather maps and charts of movements of ice fields.

FTD/SNAP  
CSO: 375

UDC 681.324.068

## COLLECTIVE USE COMPUTER SYSTEM FOR THE ACADEMY OF SCIENCES KaSSR

Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 2, Mar-Apr 1985  
(manuscript received 23 May 84) pp 108-111

[Article by U. M. Sultangazin, I. T. Pak, M. A. Bulakh, G. T. Manabayev and V. A. Tsay under the rubric: "Automation of Scientific Research"]

### [Text] Introduction

At this time electronic computer technology is penetrating many branches of industry at various levels and is solving increasing complex problems. A number of automated control systems for production and engineering processes have been created and are being developed to solve these types of problems. The role and place of the computer in automated control systems has quite definitely grown because of this.

A particularly promising field for computer application are automated systems for scientific research which are oriented toward automating labor-intensive processes and which facilitate the establishment of principles behind the phenomena studied. The critical need for computer technology in scientific research has been clearly pointed out by the results of investigating KaSSR Academy of Sciences institutes [1].

The use of computers in solving scientific problems can be classified as follows:

- Automation of scientific and technical calculations
- Automation of experiment data acquisition and processing
- Development of mathematical models for processes investigated
- Automation of scientific and administrative task control at the Academy of Sciences KaSSR
- Automation of user information servicing

Scientific/technical calculation and mathematical modeling automation includes solving the most complex problems and debugging user programs which primarily require general-purpose computer software. A number of Academy of Sciences KaSSR institutes are already using finished program systems or are tending toward these. Subscribers with this type of task to perform are most suitably handled by providing them with access to the computing power of a YeS computer

operating in the batch and/or interactive modes. Experiment data acquisition and processing tasks often require computations to be structured on a real-time basis in accordance with the process being studied. This often necessitates a physical connection between the experiment apparatus and the computers. This type of function can most suitably be executed on computer measurement systems which include an SM-4 (or SM-3) computer and an interface device (ASET or CAMAC). Use of a computer measurement system with a microcomputer assures the formation of more effective scientific research automation systems. The final processing of experiment data, however, often requires the computational resources of general-purpose computers. The automation of scientific and administrative task control and user information servicing also require powerful computers and the organization of remote access to these.

Thus, the development of scientific research using computer technology at the Academy of Sciences KaSSR requires the creation of a collective use computer system (the Academy of Sciences KaSSR VSKP). The collective use of computer technology had to be assured at minimal expense to subscriber needs for computational and informational resources. Creation of the VSKP will serve as the basis for the extension of scientific research automation systems to Academy of Sciences KaSSR sections.

The creation of the VSKP is intended to increase scientific research efficiency at the Academy of Sciences KaSSR through the use of modern mathematical methods and computer technology.

#### VSKP structure

Traditionally, collective use computer systems have a centralized structure in which the primary computational resources are concentrated in one place and remote users interact with the system via terminals. This structure can be implemented on the basis of mass-produced hardware and software, but it has the following disadvantages:

- A low level of reliability; a failure of either the central computer or the communications line to the remote terminal will deny the user access to the computer facility.
- The use of powerful computer resources in an interactive mode for preliminary operations (establishing communications, formulating assignments, etc.); this sharply reduces system performance in executing fundamental computations.

The above shortcomings are successfully overcome in a hierarchical structure in which powerful computers are located at the center of the information computer system while small units and microprocessors are placed near the periphery as an interface with outside equipment and to perform preliminary processing of user tasks. In this type of architecture, the VSKP acquires features characteristic of a computer network [2] which provides the most rational decentralization of computer resources. However, the use of various types of hardware in this case makes proper interaction of various computers and program systems essential.

An analysis of trends in computer development and utilization [2,3,4] shows that collective utilization in computer networks is becoming the dominant form of computer use. These networks are implemented with standard software or with program systems specially designed for network organization. Thus, the VSKP must be a territorially dispersed non-uniform system with a developed network of terminal equipment and communications channels. These requirements formed the basis for the creation of the Academy of Sciences KaSSR VSKP. Any Academy of Sciences KaSSR subunit with an information transmission or processing requirement can be a VSKP subscriber. Subscribers communicate with the VSKP through central and local computer complex terminals. A local computer complex uses a minicomputer (SM-3 or SM-4) to concentrate subscribers along geographical lines. At this time there are five such centers operating in a stand-alone mode. The number and power of local computer complexes will grow in the future. Among the functions to be provided by the local computer complexes are: reception and preliminary (or final) processing of information from subscribers, as well as the transmission of such data, when required, via a communications computer to the central computer complex or to other local computer complexes.

The primary computational and informational power will be concentrated in the central computer complex, currently equipped with YeS 1022 and YeS 1045 equipment and capable of future expansion. Functionally, the central computer complex provides efficient computation and processing of informational data in a local and remote batch-processing mode.

The VSKP structure provides for access to individual central computer complex hardware and software systems via a communications computer or by means of local terminal systems. The communications computer takes the form of a data transmission system which will be created during later stages of VSKP development using the packet-switching method. The possibility of interfacing the VSKP with the Akademset' [Academy Network] is envisioned [4].

The Academy of Sciences KaSSR VSKP is designed and implemented using series-produced domestic equipment: YeS and SM computers; A711-18 interface equipment; A723-5/1 and BSADS-2 communications adapters; and YeS7920 local and remote workstations.

The following basic principles were observed in designing the Academy of Sciences KaSSR VSKP [3].

Distributed processing: Creation of the central computer complex and several local computer complexes during the initial stages of system development allows the possibility of organizing an application software architecture that will assure the most efficient operation of the system as a whole. Thus, local computer complexes equipped with computer measurement units could be converted to experiment automation systems allowing real-time processing of data from experiments. Further, local computer complexes which include the OS RV [real-time operating system] will be able to debug program systems in an interactive mode as well as calculate a number of problems. The powerful resources of the central computer complex are intended for use both in the batch mode and in an interactive mode with local and remote YeS7920 terminal units.

**Specific use and specialization of computer facilities:** This principle assures rational operation of VSKP elements. Thus, minicomputers, capable of real-time operations, are assigned to handle local computer center and communications processor functions. Central computer complex equipment provides subscribers with the fundamental resources required for solving complex problems: adequate internal and external memory capacity, high speed and highly developed software.

**Standardization:** Provision is made for the use of standard YeS and SM computer software as well as for satisfying interprogram interface requirements through the adaptation of special programs. The standardization principle for informational operations calls for the use of all-union systems of information coding and classification and unified documentation systems. A collection of Academy of Sciences KaSSR algorithms and programs is operating and under development in the VSKP.

A number of other principles [3] were used in the VSKP's design, with one of these being the provision of a capacity for future expansion of the system.

#### **System-wide central computer complex software**

The chief characteristics of scientific research tasks carried out with the assistance of computers are a high percentage of time spent in the development and modification of programs, computations using programs which require a great deal of processor time, and the low relative significance of fixed tasks. These characteristics call for two basic operating modes on the part of central computer complex equipment, the time-sharing mode and the batch-processing mode.

Most of the equipment runs in the time-sharing mode during daytime hours, thus users can interactively write and debug programs on local and remote terminals. In the batch-processing mode, used by the majority of equipment during nighttime hours, user tasks are executed by the finished program without interruption.

A large number of users occupied with scientific research are familiar with FORTRAN to one degree or another and write their own programs. This situation requires the VSKP to have a developed FORTRAN programming system, a vast collection of algorithms and a highly reliable means of storing user libraries, files and utilities. Furthermore, a high-level terminal network requires easy access to finished programs on the part of users who are not data processing professionals.

In summary of the above conditions, we will formulate the basic requirements for system-wide software at the central computer complex.

1. Use of multiprogramming and computer resource utilization planning to provide a high level of system performance.
2. Provision of simultaneous access to the computer by multiple users in an interactive mode; incorporation of interactive languages (BASIC) and interactive dialects of popular programming languages (FORTRAN, PL-1).

3. Support of a vast collection of algorithms and programs, as well as easy access to these for both programmers and non-professional users.
4. Automated accounting of computer resource utilization.
5. Magnetic tape archives for reliable storage as well as timely copying and recovery of user libraries and files.

These requirements and the primary orientation of using standard software served as the basis for central computer complex system software using the YeS operating system supplemented with well-known and widely used systems and application program packages such as KROS, RRV, ORGVYTs, DISP and FORTRAN.

In addition, two packages were developed to satisfy requirements 3 and 5 in the Academy of Sciences KaSSR VSKP: the automated algorithm and program collection (AFAP) and the data set protection and servicing system (ZOND).

The VSKP user must be provided with real-time means of obtaining information on the availability of application software, specific algorithms and programs, as well as information on the use of one algorithm or another, etc. It is apparent that existing hard-copy means of information support, inconvenient even on ordinary computers, is absolutely unsuitable in the VSKP with its remote access features. The following functions are to be carried out in the AFAP system:

- Provision of information for programmers on the program and algorithm collection, based on scientific program packages.
- Uniform display of finished programs in the collection and easy access to these programs by non-professional personnel.

The program complex is designed for use in the RRV YeS operating system environment.

One of the fundamental engineering problems in the VSKP is the rational and reliable use of external memory, thus the ZOND system will have the following goals:

- Minimizing the total amount of disk memory used
- Timely deletion of obsolete and non-cataloged data sets
- Automation of the magnetic tape backup function (timely copying and recovery, maintenance of record and media catalogs, etc.)

#### Network data teleprocessing software resources

In the selection of network teleprocessing software and in the selection of VSKP hardware the developers strove for the use of standard, centrally developed and supported systems. The real-time network teleprocessing program package (PP/STC/RV) developed at the INEUM [Institute for Electronic Control Machines] in Moscow is one such system chosen to control local computer complex minicomputer operations. The PP STC/RV package expands SM computer real-time operating system and real-time distributed operating system

capabilities in the area of organizing interaction between machines. The use of the package's network maintenance resources and servicing utilities permits:

- Message transmission between terminals in the various network nodes
- Access to files in any network node
- Control of program execution at any network node
- Creation of user program systems which employ the resources of two or more computers in the network.

A four-level protocol system is used to perform these functions. Each of the protocols is a separate authorization for controlling the format, sequence and means of message transmission.

At the user level, a subscriber or applied task interacts with the batch servicing facilities to carry out the actions required for information transmission and task or file control at remote nodes in the computer network. This is the only PP STO/RV structure level visible to the user. All other actions for controlling system logic/data channels and data transmission hardware are carried out automatically.

At the logic channel control level, the communications session control protocol allows several logic channels to share a single physical communications line. At this level, user tasks with the assistance of macroaids and batch subprograms access network resources, establish communications and transfer data between their components located in different network nodes.

At the data channel control level, the system provides data transmission, error monitoring and protection against errors caused by physical lines and communications devices. The physical-level software consists of data transmission hardware drivers.

The current PP STO/RV version provides data transmission between two network nodes and message routing within the system cannot take place. The inclusion of routing facilities and an interface with other computer networks is proposed for a future version of the package.

An INEUM-developed package of local, multimachine SM and YeS computer complex organization programs (PP MMK/L) is used to provide an interface between the communications processor and central computer complex operating systems. The package provides a single-channel communications interface for data exchanges between processes running on SM and YeS computers. At the macroinstruction level of intermachine exchanges, the package emulates a YeS7906 terminal algorithm on the SM computer. This is supported by a graphic access method on the YeS computer and provides access to SM computer YeS operating system components which use this method (DUVZ, KAMA, etc.).

A network remote task input program system (SSUVZ) is being developed to organize central complex YeS computer interaction with the minicomputer network. This system's components operate in the real-time operating system and PP STO/RV environments on the communications processor and the

minicomputers at the peripheral centers for automating scientific research. User assignments, prepared using operating system utilities on peripheral computers, are transmitted by the network remote task input system to the central computer complex YeS units via the communications processor and the local multimachine complex organization program package.

Teleprocessing resources permit the greatest efficiency in the use of central complex equipment while allowing the user to work at a lower system level on a real-time basis using interactive systems in the preparation of data.

#### BIBLIOGRAPHY

1. Sultangazin, U. M. and Pak, T. I. "Sozdaniye vychislitel'nogo tsentra kollektivnogo pol'zovaniye Akademii nauk KaSSR" [Creation of a Collective Use Computer Center for the Academy of Sciences KaSSR]. VESN. AN KaSSR, 1980, No 7, p 15-20.
2. "Sravnitel'nyy analiz ryada krupnykh zarubezhnykh setey EVM" [Comparative Analysis of a Number of Large Foreign Computer Networks]. V. M. Glushkov, D. I. Nikolenko, A. A. Stogniy, et al. USiM, 1975, No 5, p 1-12.
3. Marchuk, G. I. "Territorial'no raspredelenny mnogomashinnyy vychislitel'nyy tsentr kollektivnogo pol'zovaniya SO AN SSSR" [Territorially Distributed Multi-Machine Computer Center of the Siberian Department of the Academy of Sciences USSR]. Novosibirsk, 1980, 58 pp. (Preprint/Computer Center, Siberian Department, Academy of Sciences USSR).
4. Yakubaytis, E. A. "Arkhitektura vychislitel'nykh setey" [Computer Center Architecture]. Moscow: Statistika, 1980, 279 pp.

COPYRIGHT: Izdatel'stvo "Naukovo Dumka", "Upravlyayushchiye sistemy i mashiny", 1985

12746  
CSO: 1863/270

"NEW" MOTOR FOR ROBOTS DESCRIBED

Moscow SOTSIALISTICHESKAYA INDUSTRIYA in Russian 16 May 85 p 4

[Article by I. Demchenko: "Tests are Proceeding: New Robot Muscles"]

[Text] Scientists at the VNIIElektroprivod [All-Union Scientific, Planning and Design Institute for Automatic Electric Drives in Industry, Agriculture and Transportation] proposed that the conventional three-phase asynchronous electric motor be used in robot equipment. We produce these motors by the millions in this country and they have already gone through their initial check on the welding manipulator at the Institute of Electric Welding imeni Ye. O. Paton. Tests are now continuing and robot-assembler hands will soon get new "muscles".

The attentive reader might object that these motors have been available for some time and might want to know what is new in the VNIIElektroprivod specialists' idea. And if they are being used extensively in industry, why are more tests necessary? We asked the manager of the developer group and Candidate of Technical Sciences V. Mishchenko to answer these questions.

"There only seems to be a paradox here," says Vladislav Alekseyevich. These motors have not been used in robot equipment. Usually mechanical helpers are equipped with special direct-current electric motors, and for good reason. They are almost always preferred when the speed and load have to be regulated. The fact is that they have a more complex design, but in return their control is very easy and they follow commands quickly and accurately. But these motors are not cheap.

And after all, a robot doesn't need just one drive. It requires as many drives as the mechanical arm has articulators. Manipulators used in contemporary industry must have six to nine degrees of movement and experimental models have already been developed with 18 points of motion and so more and more motors are required.

Certainly one shouldn't use money as the gauge for measuring robots' use.

Robots aren't bothered by noise, heat or dust, are not fatigued by monotonous work and are used to replace people in dangerous production. But one cannot discount economics, especially since robotization is one of the national trends in scientific-technical progress.

Here is why it would be tempting to place one of the simplest, most widely distributed and cheapest electric motors in production into use in robot equipment. Until now they have been considered unfit for delicate work because of their poor accuracy at work and the complexity of their control. We have succeeded in controlling these "obstinacies" by developing fundamentally new control methods. This allowed us to put in robots a motor that man has been using for about one hundred years now. They are one-tenth the price and three times more reliable than contemporary "muscles" in our mechanical helpers. Labor consumption in producing the asynchronous motor is much less than that for a direct current motor and their specific horsepower is greater.

I will not go into the delicacy of the asynchronous motor's control system as it is rather complicated. I will note only its compactness. The integrated microcircuits and power transducer are in a block consisting of two printed circuit boards that are 20 by 30 centimeters. One block controls the work of one motor and a cassette of six blocks controls an entire "mechanical arm". The cassette is combined with the robot's "brain", a microprocessor system or a micro-computer. According to preliminary calculations, each six-coordinate electric drive can save the national economy at least 30,000 rubles.

The initial tests have already shown that the new asynchronous motor control system in the "joints" of mechanical arms have substantially increased the robot's speed of action and the accuracy of its motion.

12511  
CSO: 1863/326

UDC 519.685.4:62.5+007.52

## ALGORITHMS FOR REPRODUCING TOOL MOVEMENT PATHS FOR PROGRAM-CONTROLLED EQUIPMENT

Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 2, Mar-Apr 1985  
(manuscript received 18 Oct 83) pp 114-116

[Article by V.M. Vodovozov under the rubric: "Experience in the Development and Introduction of Automated Control Systems"]

[Text] As a rule, problems of tracing tool movement or robot grip paths in modern numerical control (NC) systems are solved in two steps: path approximation during program preparation and approximation curve interpolation in a real-time mode. Approximation and interpolation algorithm development is a critical process in implementing the flexible and precise level of control required in modern machining production applications.

According to Weierstrass' theorem [1], each function  $f(x)$  that is continuous on a given interval can be approximated by a polynomial  $p(x)$  such that  $|f(x) - p(x)| < \epsilon$ , where  $\epsilon$  is any positive number. The simplest and most frequently used polynomials in NC systems are described by a straight line and a circle. Second- and third-degree polynomials are used in special applications, if a majority of the surface processed has a random form, as in the production of vehicle bodies, aircraft wings, and blades for helicopter rotors or ships' screws. As the approximation precision rises, the complexity of interpolation algorithms increases.

Let us assume a given curve  $f(x)$  approximated by polynomial  $p(x)$  with a root-mean-square error  $\epsilon_1(x)$  after which the approximation segments are subjected to interpolation with an error of  $\epsilon_2(x)$ . If the interpolation algorithm is introduced in the recurrent approximation relation:

$$\left. \begin{array}{l} x_{i+1} = x_i + u(x_i, y_i, t_i) \Delta t \\ y_{i+1} = y_i + v(x_i, y_i, t_i) \Delta t \end{array} \right\} . \quad (1)$$

then the resulting introduced error in (1) can be evaluated using Taylor's expansion:

$$\left. \begin{aligned} x_{i+1} &= x_i + x'_i \Delta t + \frac{x''_i}{2} \Delta t^2 + \dots + \frac{x_i^{(n)}}{n!} \Delta t^n \\ y_{i+1} &= y_i + y'_i \Delta t + \frac{y''_i}{2} \Delta t^2 + \dots + \frac{y_i^{(n)}}{n!} \Delta t^n \end{aligned} \right\}, \quad (2)$$

where  $x_i, y_i$  — are the current curve coordinates,  $\Delta t$  represents time and  $u$  and  $v$  are some functions.

According to (2), the coordinates of the next point  $(x_{i+1}, y_{i+1})$  depend not only on the current coordinates  $(x_i, y_i)$  and their movement rates, but also on the sequences produced above. In the case of movement in a straight line, the sequences produced above are equal to zero and successive points on the path can be computed from a truncated Taylor series:

$$\left. \begin{aligned} x_{i+1} &= x_i + x'_i \Delta t \\ y_{i+1} &= y_i + y'_i \Delta t \end{aligned} \right\}. \quad (3)$$

Thus, in the case of a linear approximation  $f(x)$ , the linear interpolation error is  $\epsilon_2(x) = 0$  and the resultant error in reproduction is defined by the approximation error  $\epsilon_1(x)$  and the robot unit drives. Any more complicated approximation capable of reducing the  $\epsilon_1(x)$  error usually leads to an increase in the interpolation error in addition to complicating the system's algorithm arrangement.

We will examine linear interpolation algorithms which provide the maximum computation speed in a real-time mode with three-coordinate NC systems featuring built-in microcomputers. Considering that evaluation function algorithms [2] are characterized by a tendency to increase the error at the end of the frame ( $\epsilon_2 \neq 0$ ), we will turn to a digital differentiation algorithm which provides high accuracy on reaching the end point of an approximated straight-line segment [3,4].

In parametric form the straight-line equation is:

$$x = x_0 + \int_0^t a d\tau, \quad y = y_0 + \int_0^t b d\tau, \quad z = z_0 + \int_0^t c d\tau, \quad (4)$$

where  $x_0, y_0$  and  $z_0$  are the initial coordinates, and

$$a = \frac{dx}{d\tau}, \quad b = \frac{dy}{d\tau}, \quad c = \frac{dz}{d\tau}$$

are the coordinate movement rates.

Since  $a, b, c = \text{const}$ ,

$$\left. \begin{aligned} \int_0^t a d\tau &= \sum_{i=1}^N a \Delta \tau_i = \sum_{i=1}^N (A/N)_i \\ \int_0^t b d\tau &= \sum_{i=1}^N b \Delta \tau_i = \sum_{i=1}^N (B/N)_i \\ \int_0^t c d\tau &= \sum_{i=1}^N c \Delta \tau_i = \sum_{i=1}^N (C/N)_i \end{aligned} \right\}. \quad (5)$$

where A, B and C are frame increments by coordinates,  $t_i$  is the processing time for the  $i$ -th cycle,  $N = 2^n$ , and  $n$  is the sequence number of the most significant bit in the largest increment.

Addition in (5) continues until all A, B, C segments are covered. Whole portions of the totals are passed on for processing by the unit's drives. The rate of addition, defined by the required speed of reproduction, is computed at the program preparation stage for each frame. Figure 1 illustrates the algorithm used to implement (4) and (5).

The frame change frequency, which must be maximized in a linear approximation of paths, depends on the dynamic capabilities of the unit's drives and the speed with which the algorithm is processed in the interpolator. The boundaries imposed by this last factor are often the basis for choosing circular interpolation in the case of movement along curves and lines; this of course complicates shaping algorithms when processing straight lines.

The computation speed achieved with this algorithm can be increased if one of the coordinates, along which movement in the frame is the greatest, is used as a guide, as often occurs in evaluation function algorithms [2]. In this case, a control pulse will be sent to the guide coordinate during each interpolation cycle while control of the guided coordinates takes place as a function of the guide coordinate.

Assuming  $N = Z$  in (5), we will attempt to calculate the current values of guided coordinates  $x_i$ ,  $y_i$  by multiplying the current code value of guide coordinate  $z_i$  by the factors  $A/Z$  and  $B/Z$ . These latter can be calculated when preparing the frame, while a microcomputer's extended arithmetic instruction or hardware multiplier can be used for real-time multiplication.

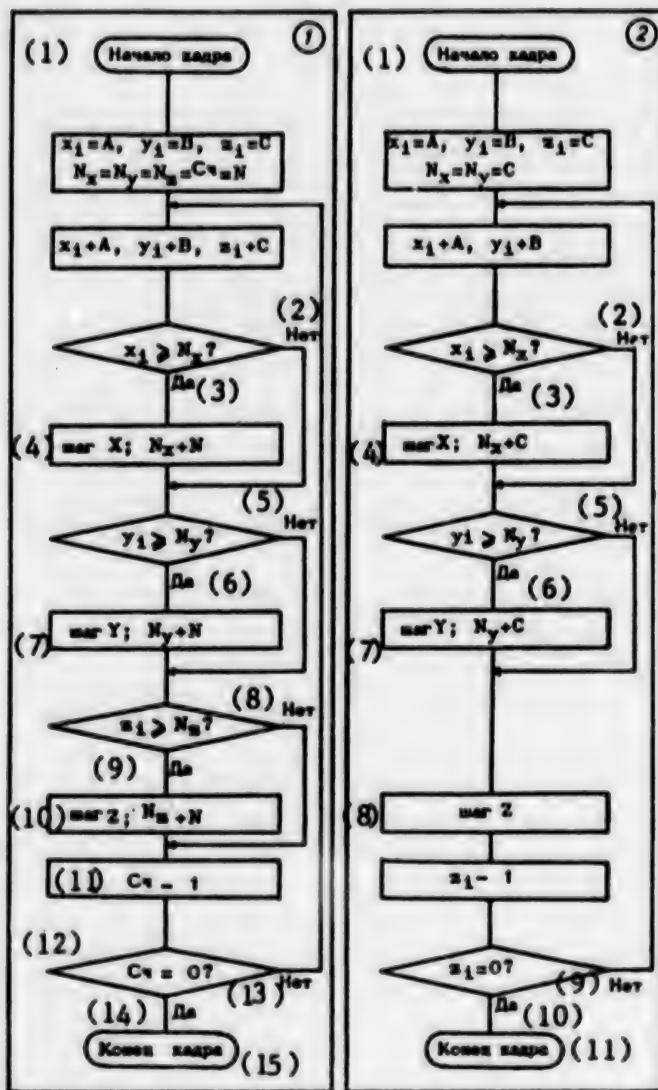
The exceptional simplicity of such an approach does not compensate for its significant drawbacks: slow multiplication speed and low calculation accuracy due to the computer's finite word length, a condition graphically demonstrated in division operations. A more accurate, albeit slower, algorithm takes the form:

$$\left. \begin{array}{l} x_i = (z_i A) / C \\ y_i = (z_i B) / C \\ z_{i+1} = z_i + 1 \end{array} \right\} \quad (6)$$

with the execution of a step along X upon  $\text{ent}(x_i) > \text{ent}(x_{i-1})$ , a step along Y upon  $\text{ent}(y_i) > \text{ent}(y_{i-1})$  and a mandatory step along Z during each cycle, because in this case at the segment ends  $\text{ent}(x_i) = A$  and  $\text{ent}(y_i) = B$ .

The low execution speeds for multiplication and division operations in (6) are the reason for preferring algorithms (analogous to that in Figure 1) based on addition and shift operations, even when a guide coordinate is present.

According to Figure 2 (in which Z is the guide coordinate), the interpolation algorithm with a guide coordinate not only has a shorter computation cycle, it generally has a lower number of cycles ( $C \leq N$ ) in comparison with the base algorithm (Figure 1). Thus, in an erroneous count of a control example using a



- (1) Start of frame
- (2) No
- (3) Yes
- (4) Step X,  $N_x + N$
- (5) No
- (6) Yes
- (7) Step Y,  $N_y + N$
- (8) No
- (9) Yes
- (10) Step Z,  $N_z + N$
- (11) Counter - 1
- (12) Counter = 0?
- (13) No
- (14) Yes
- (15) End of frame

Fig. 1 - Base interpolation algorithm

- (1) Start of frame
- (2) No
- (3) Yes
- (4) Step X,  $N_x + C$
- (5) No
- (6) Yes
- (7) Step Y,  $N_y + C$
- (8) Step Z
- (9) No
- (10) Yes
- (11) End of frame

Fig. 2 - Interpolation algorithm with guide coordinate

frame with parameters of  $A = 9$ ,  $B = 2$  and  $C = 11$ , an Elektronika-60 microcomputer spends 1200 ms using the base algorithm (see Figure 1) in 16 cycles, 3700 ms using algorithm (6) in 11 cycles and 700 ms using the algorithm in Figure 2 in 11 cycles.

The need to define supplementary data (the length of word N in the base algorithm and the sign of the guide coordinate in the guide coordinate algorithm) when writing the program and transmit this information to each frame is a disadvantage of both algorithms. This can be overcome in profile systems of NC units with continuous coordinate movement (without stops at even one of the coordinates). These systems include thread-cutting units, winding machines and painting robots. One of the coordinates in such a system can be permanently fixed as a guide, if its resolution is chosen in such a manner as to provide that its frame increment always exceeds the individual coordinate increments. This eliminates the operations involved in guide coordinate selection and algorithm modification during frame changes.

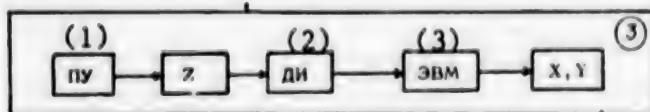


Fig. 3 - Fixed guide coordinate control system

Figure 3 shows a block diagram of the engineering solution used in the SKAN-4 NC system for a large winding machine. The operator uses the control console (1), to input the rotation rate of the spindle's main drive (Z). Regulated only by speed, the main drive unit also turns an attached pulse sender (2). Each pulse from the sender is latched in a microcomputer (3) which solves an interpolation problem and sends control pulses to the drives at the guided coordinates (X,Y). An Elektronika-60 microcomputer can execute the algorithm's operation cycle at a rate of 10 kHz. The maximum spindle rotation rate is 16 rpm and the pulse sender's frequency is selected by an electronic reduction unit within a 0.1-0.667 kHz range. During interpolation-free periods (not less than 900 ms during each cycle), the microcomputer controls unit coordinate relative movement indications or executes monitoring functions requested by the operator.

Software result analysis and operational experience have indicated that the use of linear interpolation algorithms based on digital differentiation with a fixed guide coordinate provides the best speed and reproduction accuracy characteristics for spatial program paths.

#### BIBLIOGRAPHY

1. Smirnov, V. S. "Kurs vysshey matematiki" [Higher Mathematics Course]. Moscow: Nauka, 1967, 656 pp.
2. Tormyshev, Yu. I. "Metody i sredstva formirovaniya shagovykh trayektoriy" [Methods and Means of Producing Step-by-Step Paths]. Minsk: Nauka i tekhnika, 1980, 142 pp.

3. Papaioannou, S. J. Interpolation Algorithms for Numerical Control. COMPUT. INDUSTRY, 1979, v 1, No 1, p 27-40.
4. Martynichev, A. K. "Algoritm formooobrazovaniya v sisteme chislogo programmnogo upravleniya tipa CNC" [Shaping Algorithm in a CNC System]. "Elektrooborudovaniye promyshlennyykh predpriyatiy" [Electrical Equipment in Industrial Facilities], 1981, 9th ed, p 85-89.

COPYRIGHT: Izdatel'stvo "Naukovo Dumka", "Upravlyayushchiye sistemy i mashiny", 1985

12746  
CSO: 1863/270

UDC 681.3:744

## MINIMIZING IMAGE FORMATION TIMES ON A DRAFTING/GRAFIC AUTOMATON

Novosibirsk AVTOMETRIYA in Russian No 2, Mar-Apr 85 (manuscript received 16 Nov 84) pp 99-100

[Article by V. V. Kravchuk (Kuybyshev) under the rubric: "Brief Reports"]

[Text] The operational efficiency of many existing automated design systems depends to a great extent on graphic data output device performance, particularly that of drafting/graphic automata. The relatively low throughput of these units markedly reduces an entire system's efficiency, creating long delays in the process of obtaining drafting/graphic hard-copy output.

This report examines the problem of minimizing idle movements of the plotting head during image formation on a pen-type drafting/graphic plotter.

We will assume that the drawing is made up of a set of segments on a plane drawn from one end point to another (the condition for continuous drawing of segments is linked to the quality demands imposed on the image formed). We will form a graph model of a drawing. For this, we will place the drawing's segments in line with the edge of a complete pair combination  $M$ . We will expand the pair combination formed into a completely undirected graph  $G(X)$ , in which  $X$  is the set of pair combination vertices. We will enumerate the  $X$  vertices so that condition

$$(2i-1, 2i) \in M, \quad i = \overline{1, N}.$$

is fulfilled. Here,  $N$  is the number of edges in pair combination  $M$ .

We will use  $c(i, j)$  to designate the length of the segment connecting point  $i$  with point  $j$ . Now we will formulate the problem of minimizing the number of idle movements of the plotting head in the following manner: find the set of edges  $U$  in graph  $G(X)$  which minimizes

$$\sum_{(i,j) \in U} c(i,j) \rightarrow \min. \quad (1)$$

and satisfies the limitation

$$G_1(X, M \cup U) \text{ — an Euler graph.} \quad (2)$$

Assuming the number of idle movements to be minimal, it is easy to show that (2) can be exchanged for one of the following conditions:

- a)  $G_1(X, M \cup U)$  -- Hamiltonian cycle
- b)  $G_1(X, M \cup U)$  -- alternating cycle
- c)  $G_1(X, M \cup U)$  -- cycle

From a) and b) it follows that  $U$  is a complete pair combination and  $|U| = |M|$ . Thus, the problem of minimizing idle movements is equivalent to the traveling-salesman problem with partially fixed transients forming the complete pair combination  $M$ .

The known algorithms for accurately solving the traveling-salesman problem have an exponential time complexity [1] which precludes their use for practical purposes. Therefore, a heuristic algorithm, based on a dynamic programming method and allowing parallel computation, is suggested for solving this problem. The algorithm's time complexity is  $O(N^4)$  and its space complexity is  $O(N^2)$ .

Let us examine the algorithm proposed. In describing it we will use the term "alternating circuit" [2] to designate the circuit in which one of each pair of adjacent edges belongs to the pair combination  $M$ , while the other does not.

Step 1. Select vertex  $i$  and edge  $(i, j) \in M$ ; let  $m = 3$ .

Step 2. Form an alternating circuit  $S(i, k, m)$  with a length of  $m$ , connecting vertex  $i$  with the remaining vertices and beginning with edge  $(i, j)$ .

Step 3. Increment  $m$ :  $m = m + 2$ .

Step 4. Form a set of alternating circuits with a length of  $m$  such that the initial segment, consisting of the first  $m - 2$  edges of each circuit, coincides with one of the circuits  $S(i, k, m - 2)$  with a length of  $m - 2$ .

Step 5. From among the alternating circuits formed, with a length  $m$  and ending at the same vertex  $k$ , select the one with the minimum weight and designate it  $S(i, k, m)$ .

Step 6. If  $m < 2N - 1$ , branch to step 3.

Step 7. Close each alternating circuit  $S(i, k, m)$  in the cycle, completing one edge. Select the cycle with the lowest weight from among the cycles formed.

To eliminate the formed cycle's dependence on the initial vertex, the sequence of steps 1-7 is executed beginning with each vertex in  $X$ . The best of the cycles formed in this manner is then used to solve the problem.

This algorithm was used as the basis for a FORTRAN-IV program to solve a number of testing problems involving initial data that varied in accordance

with a statistical principle. The number of segments ranged from 5 to 25. Calculation results showed that after optimization the length of idle movements was reduced three-fold on average. On the basis of the computations carried out, one can conclude that the result of the first iteration (steps 1-7) differs from that of the final iteration by no more than 10 percent. In a number of practical applications this allows a one-iteration limit to be imposed, for a  $2N$ -fold reduction in calculation time.

Note must also be made of the fact that accurate solutions are obtained when the initial image can be drawn without raising the plotting head.

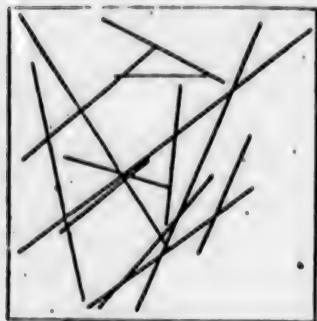


Fig. 1

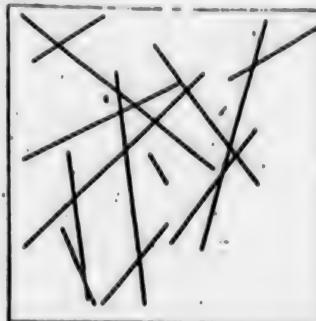


Fig. 2



Fig. 3

Figure 1 shows one of the test images consisting of 13 segments arranged on a 100 x 100-mm square. Figure 2 shows random idle movements. Figure 3 shows the results of the idle movement minimization algorithm's operation. This problem was solved in 2.7 minutes on an SM-4 computer. The random idle movement length was 646.43 mm. After optimization, the length of random movements was 186.88 mm.

The algorithm examined in this article can also be used in programming numerical-control machine units to minimize the idle movements of a cutting tool.

#### BIBLIOGRAPHY

1. Aho, A., Hopcroft, J. and Ullman, J. "Postroyeniye i analiz vychislitel'nykh algoritmov" [The Design and Analysis of Computational Algorithms]. Moscow: Mir, 1979.
2. Maynika, E. "Algoritmy optimizatsii na setvakh i grafakh" [Optimization Algorithms in Networks and Graphs]. Moscow: Mir, 1981.

COPYRIGHT: Izdatel'stvo "Nauka", "Avtometriya", 1985

12746  
CSO: 1863/355

UDC 621.372.542

## DIGITAL PROCESSING OF IMAGES IN TRACKING A MOVING OBJECT

Leningrad PRIBOROSTROYENIYE in Russian No 2, Feb 85 pp 39-43

[Article by B. A. Alpatov, A. A. Selyayev and A. I. Stepashkin. Ryazan Radiotechnical Institute]

[Text] A digital algorithm for digital processing images in tracking a moving object is proposed. The results of computer simulation are given which confirm the expediency of using the preliminary smoothing of images to increase the precision of evaluating coordinates.

In creating certain varieties of industrial robots, "imparted with vision" [1], problems arise related to the development of algorithms for tracking moving objects. Most of these algorithms are correlational and are based on comparing sections of a current image with some previously obtained reference image of the object [2]. Considering the great requirements expected of a high speed algorithm, the measure of difference is used frequently

$$F[v, \mu] = \sum_{i=1}^M \sum_{j=1}^N \left| g[i + v - \frac{M}{2}, j + \mu - \frac{M}{2}] - h[i, j] \right|. \quad (1)$$

where  $v, \mu = \frac{M}{2}, N - \frac{M}{2}$  are the number of the line and the number of the column of matrix G the size of  $N \times N$  of the current images;  $g[i, j]$  are elements of matrix G,  $h[i, j]$  are elements of reference image H with a size of  $M \times M$ ,  $N > M$ .

Values  $v=v^*, \mu=\mu^*$ , at which  $F[v, \mu]$  reaches a global minimum, are assumed as the coordinates of the center of the object being tracked at the current frame of the image.

When tracking a moving object which changes its shape with time (for example due to turns), or when the light changes, it is obviously necessary to change the reference image at certain moments of time. In this case, as a new reference, a part of the current image may be selected with center coordinates, calculated according to (1). However, due to interferences that distort the image, the quantization noises or discretization errors, the determination of the coordinates is done with some error which is introduced in the newly-formed reference. As a result of this, when the reference image is changed many times,

the error dispersion will increase with time, leading in the end to the tracking failure. This is especially true when the reference signal is changed with each frame when the object moves slowly. It was established experimentally that there may be no tracking at all in this case.

A shortcoming of other approaches [2], in which the reference image is not used, is the low noise resistance under considerable noise conditions and the presence of extraneous objects in the images.

The consideration of the noted circumstances leads to the necessity of modernizing the correlational algorithms so that errors, originating when the reference image is changed, do not accumulate. The idea of the proposed approach is based on the fact that the object is moving and that at the initial moment of time, it is thought to be the known reference image. A block diagram of the corresponding algorithm is shown in Fig. 1.

According to expression (1), coordinates of the position of the object  $v_n^*$ ,  $\mu_n^*$  are determined on the current frame of the image. In the case of satisfactory correspondence between the reference image and the found object, an  $M \times M$  section of the image, containing the object, is smoothed according to a known algorithm of current averaging [3].

$$v_n[i, j] = (1 - \beta) \cdot g_n \left[ i + v_n^* - \frac{M}{2}, j + \mu_n^* - \frac{M}{2} \right] + \beta \cdot v_{n-1}[i, j]. \quad (2)$$

where  $v_n[i, j]$  is the smoothed value of the signal;  $n$  is the frame number;  $\beta$  is the parameter of exponential smoothing;  $i, j = 1, M$ . The result of such processing is a reduction in the dispersion of additive uncorrelated noise by  $\frac{1}{1+\beta}$  times. Another positive consequence of temporary averaging of signals  $1 - \beta$  from the object, separated from a sequence of frames, is that due to the tracking of the motion of the object, the contrast of extraneous stationary objects, in the area of the reference image, is reduced. A similar result is also obtained when summing signals from the object during  $L$  frames. The signal to noise ratio increases in this case  $\sqrt{L}$ -fold. This makes it possible, by further threshold processing, to separate the contours with fair assurance and, therefore, also points  $z[i, j]$ , that belong to the object of interest. Some "center" of the object can be determined with known algorithms from the separated external contour of the object.

After the "center" of the object is determined, the reference signal is replaced by a new one, "cut-out" of the current frame relative to the found center. Obviously, it is expedient to use results of temporary averaging to correct the center of the reference until the expiration of the  $L$  frames after the change of the reference signal. In the case of the current averaging  $\beta$  is selected equal to  $\frac{L+1}{L-1}$  [3]. The number of  $L$  frames, in its turn, is selected in reverse proportion to the signal to noise ratio and the degree of mobility of the object. The determination of the moments for the expediency of the change in the reference signal is made by comparing the value of the function difference at a point of the best coincidence with some threshold value  $R$ . Depending upon

the available information, the function difference is calculated either over the area of MXM, or according to area Z, corresponding to the configuration of the object. By neglecting the error in evaluating coordinates caused by noise and by discretization effects, it is not difficult to show that, in the case of the additive, uncorrected, gaussian noise with a zero average, threshold value R can be determined as

$$R = C \sqrt{D[F_1^*]} + M[F_1^*]. \quad (3)$$

where  $D[F_1^*]$ ,  $M[F_1^*]$  are respectively the dispersion and the mathematical expectancy at the point of the best coincidence, depending upon the noise dispersion and the number of points, contained in area z;  $C = \Phi^{-1}(1 - P_{\text{err}})$ , where  $\Phi$  is the probability integral;  $P_{\text{err}}$  is the probability of erroneous change of the reference. In this case, it is assumed that the number of points contained in the reference is sufficiently high. The described algorithm makes it possible to eliminate the effect of accumulated errors inherent in the purely correlational method while preserving its advantages.

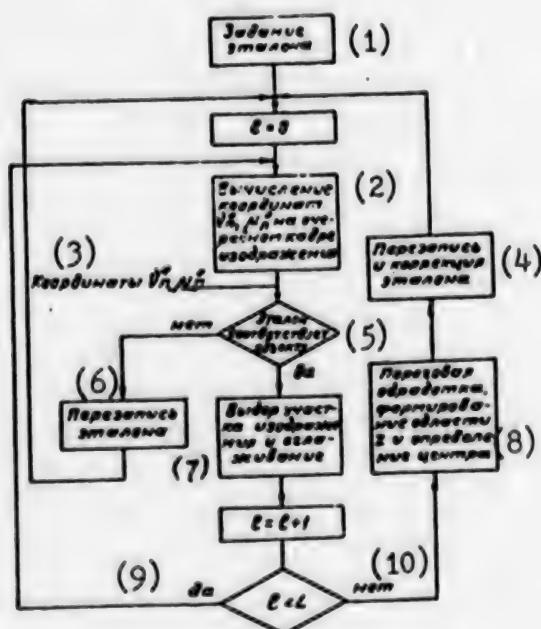


Fig. 1. 1 -- reference assignment; 2 -- calculation of coordinates for the next in turn frame of image; 3 -- coordinates  $v_n^*, m_n^*$ ; 4 -- rerecording and correction of reference; 5 -- reference corresponds to object; 6 -- rerecording of reference; 7 -- selection of image section and smoothing; 8 -- threshold processing, formation of area Z and determination of center; 9 -- yes; 10 -- no.

Experimental studies of the developed algorithm made by simulating it on the YeS-1022 computer, as well as partially on a model of a specialized computer on a real time scale, confirmed its efficiency in various situations. The operating precision of the described algorithm is determined to a considerable degree by the precision of the evaluating algorithm (1), which is close to the optimal in the sense of a criterion of maximal probability in a case of a noiseless reference signal. However, in the proposed algorithm, the reference is formed from the current noisy image. Taking that into account by introducing a preliminary two-dimensional filtration of the image, it will be possible, in a number of cases, to achieve higher precision in evaluating the position of the object at the stage of calculating the global minimum (1).

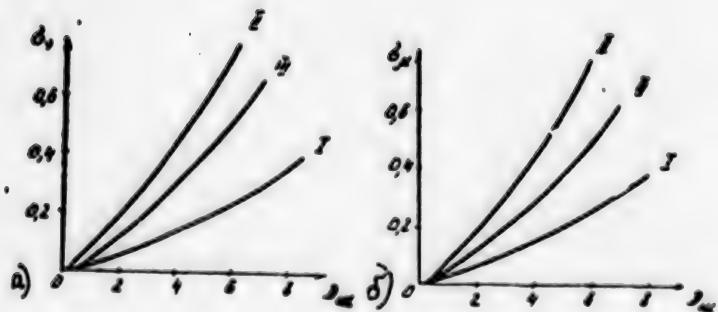


Fig. 2



Fig. 3

In confirmation of the above, results are presented of simulating algorithm (1) and an algorithm which includes the operation of the preliminary smoothing of signals by a two-dimensional filter with a weighted function

$$T = \frac{1}{9} \begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix} \quad (4)$$

Fig. 2 shows the relationships between the mean-square deviations in estimating coordinates and the dispersion of the superimposed noise for an image in Fig. 3. (On Fig. 2.: I is pure reference; II is noisy reference; III is smoothed reference). In the given example, the amplitude of the signal was selected equal to three units, the reference image (the inner square) was represented by a matrix of 16 x 16 points. The search was made in a square of 32 x 32 points. Discretization errors were not simulated. One hundred noise realizations were generated for each dispersion value. It should be noted that the selection of an optimal function of a two-dimensional filter requires apriori information on the shape of the signal.

#### BIBLIOGRAPHY

1. Katys, G. P. "Optical Informational Robot-Manipulator Systems." Moscow. Mashinostroyeniye, 1977, 272 pages.
2. Pratt, U. "Digital Processing of Images." Translated from English (edited by D. S. Lebedev). Moscow, Mir, 1982, 790 pages.
3. Korshunov, Yu. M.; Bobikov, A. I. "Digital Smoothing and Converting Systems." Moscow. Energiya, 1969, 128 pages.

COPYRIGHT: "Izvestiya vuzov SSSR, Priborostroyeniye", 1985

2291

CSO: 1863/234

## STATUS AND PROSPECTS FOR DEVELOPMENT OF GAS INDUSTRY AUTOMATED CONTROL

Moscow GAZOVAYA PROMYSHLENNOST': SERIYA AVTOMATIZATSIIA, TELEMECHANIZATSIIA I SVYAZ' V GAZOVYY PROMYSHLENNOSTI in Russian No 5, Sep 84 pp 1-29

[Abstract] Natural gas will reach 33% of the fuel-energy balance of the USSR by 1985. The main increase in production of gas from then until the year 2000 is planned for remote deposits in Western Siberia. The unified network of gas pipelines will be developed in the eleventh and twelfth 5-year plans by the creation of very high capacity gas lines forming a single, complex network, requiring basically new methods and equipment for its control. The reliability of the gas supply system will depend on the operation of a few large facilities. The mean length of gas pipelines and power expended in the transport of gas will both increase due to the movement of the raw material base into Western Siberia. Natural gas as a power supply will thus increase in cost. Given the planned scale of development of the unified gas pipeline network, improvement of the control of gas supply in the Soviet Union is possible only by use of the systems principle in development of the multilevel gas automatic control system. The development of this system, called ASU-gaz, will occur in stages, including the creation of systems for controlling operating modes and development of territorial and technological gas pipeline network subsystems. The creation of ASU-gaz requires the development of methods of hierarchical optimization, installation of a network of computers and a distributed database. ASU-gaz is intended to merge with traditional control apparatus in order to create an automated mechanism for controlling the operation and development of the pipeline network. Currently the gas industry has a branch ASU (OASUgazprom) with 12 subsystems, 23 organizational-economic ASU in production and all-union industrial associations, and 52 process control systems (ASUTP) in recovery, transport, and processing of gas. In 1983 the percentage of gas production covered by ASUTPs was 58% for recovery, 67% for transport, and 54% for processing. In 1985-1990, organizational-economic ASUs are to be unified on the basis of a branch network and distributed data base. A packet switching network which uses protocols, corresponding to international standards, is to come on line for the Urengoy-Uzhgorod link in 1986.

References 17 Russian.

[138-6508]

'ASU-INSPEKTSIYA' AUTOMATED INFORMATION PROCESSING SYSTEM

Moscow FINANSY SSSR in Russian No 10, Oct 84 pp 48-51

BUDANOV, A. G., Deputy Department Chief, The All-Union Scientific Research Institute for Automation of Control in the Nonindustrial Sphere and ORLOVA, T. D., Deputy Department Chief, Chief State Insurance Administration, USSR

[Abstract] The 'ASU-Inspeksiya' system was developed by the Institute for Automation of Control in the Nonindustrial Sphere and is oriented toward use in large State Inspection Offices where the greatest single activity is related to life insurance. It supports processing of information on life insurance agreements with some 70,000 clients, up to 130 agents, 7 inspectors' sections, up to 99 brigades, 999 inspected organizations, 99 shops and organizations. The system performs 38 jobs in four subsystems, including information support, auditing of code books and tables and checking of correctness of coding and input of information. The systems supports inquiry or printout of all forms entered into the system. 38.8 million personal accounts, or 54.5% of the total, are now processed by electronic or punch card computer. 25 million of these are processed on electronic computer, including 8.6 million using 'ASU-Inspeksiya' on YeS computers. Three software versions now exist: 'ASU-Inspeksiya-2' for YeS machines running DOS with 256 KBytes main memory and 2 7.25 MByte disks; 'ASU-Inspeksiya-MD' for YeS DOS; and 'ASU-Inspeksiya-MO' for YeS OS version 6.1. Conversion software exists, including a package from the Minsk-32.  
[205-6508]

UDC: 002:65.015.13.011.56

INFORMATION SUPPORT OF THE CENTRAL USSR STATE PLANNING COMMISSION AUTOMATED PLAN CALCULATION SYSTEM CENTRAL GROUP OF PROBLEMS

Moscow KLASSIFIKATORY I DOKUMENTY in Russian No 12, Dec 84 pp 1-6

PENCHIK, E. F., Main Computer Center, USSR State Planning Commission

[Abstract] The central group of tasks of the USSR State Planning Commission Automated Plan Calculation System (ASPR) is among the first large projects which has fully implemented the principle of integration of planning calculations in practice. The information support of this group of tasks functions on the basis of the general systems information support facility of the automated plan calculation system, consisting of more than 250 planning tasks, and also includes an independent information support system. The central group of tasks utilizes the All-Union classifiers, system-wide dictionary, rules for formal standardization of plan documents, systems for transmission of data among technical problems, albums of planning document forms and summary balance nomenclature lists. Data flows between tasks in the central task group are diagramed and described. Information

support system development in the future will include organization of flows of information in the group and tracking of transmission of data through the technological cycle of plan calculations by development of an information-technological model of planning to cover the entire central task group.

References 2 Russian.

[140-6508]

UDC: 025.4:65.015.13.011.56:681.3.01

DESIGN OF REPUBLIC STATE PLANNING COMMISSION AUTOMATED PLAN CALCULATION SYSTEM DATABASES

Moscow KLASIFIKATORY I DOKUMENTY in Russian No 10, Oct 84 pp 1-6

SOMS, R. V., Candidate of Economic Sciences, BLUM, I. A., KADISH, B. A., MARGULIS, A. M., Candidate of Technical Sciences and SHIKHMAN, Ye. S., Scientific Research Institute of Pedagogy, Latvian State Planning Commission

[Abstract] The Latvian State Planning Commission automated plan calculation system (ASPR) database is a data system organized in the form of databases plus data, software, hardware and organizational facilities designed for multipurpose utilization in processes of development of national economic plans for economic and social development of the republic. The stage of designing the database includes development of a conceptual model of the data models which must be supported by the DBMS to assure its convenient, effective utilization. During design the database is subdivided into segments, each reflected in a specific hardware-software environment. The data system is designed so that problems such as optimization, production of complex models for statistical analysis and predictions, summary balance calculations, processing of foreign trade data and administration of system-wide metadata are performed on large computers, while minicomputers undertake operational processes of monitoring and planning multivariate calculations and information-reference servicing. At the present design stage the central element of the distributed data processing system is a YeS-1055 computer. The representation of individual conceptual model elements using the UNIBAD YeS DBMS is analyzed. The approach used to reflect the conceptual model in the data logic model does not require duplication of lists of relationships as elements of the conceptual and data logic models. Report generation facilities and facilities for the construction of dialogue scenarios make the planned data model effective. Figure 1; references 5:

3 Russian, 2 Western.

[139-6508]

UDC: 025.4:64

DEVELOPMENT OF THE REPUBLIC ASSORTMENT PORTION OF THE ALL-UNION CLASSIFIER FOR INDUSTRIAL AND AGRICULTURAL PRODUCTS IN THE MOLDAVIAN SSR

Moscow KLASSEFIKATORY I DOKUMENTY in Russian No 10, Oct 84 pp 11-13

KOZINA, Z. G., KRISTYA, N. A. and YURKIV, M. V., Scientific Research Institute of Planning, Moldavian SSR State Planning Administration

[Abstract] The author's institute has developed the republic assortment segment of the All-Union industrial and agricultural product classifier, including products manufactured according to standards and technical documentation approved by the Republic State Planning Administration but not products, responsibility for the technical level and satisfaction of demands of the national economy and population for which has been given to the ministries and departments of the USSR. Development of the new standard included determination of product nomenclature to be included in the list, distribution of products according to classification groups in the All-Union classifier, sending of requests to head organizations for determination of codes or reservation of capacity, coding of products on the basis of the code series received, preparation of a draft classifier and approval of the final classifier. The system for introduction of the classifier calls for further improvement of the classifier by inclusion of names of new and previously omitted products, changes as products are changed and elimination of obsolete products. The classifier will be published twice per year.

[139-6508]

UDC: 62-52:681.3.06.2

SEMANTIC-SYNTACTIC NATURAL LANGUAGE PHRASE ANALYZER IN LEARNING DIALOGUE INFORMATION PROCESSING SYSTEMS

Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan 85  
(manuscript received 21 Dec 83; after revision 17 Apr 84) pp 80-85

DOVGYALLO, T. P.

[Abstract] Requirements are formulated for a learning dialogue system capable of converting untrained user natural language input to the internal formalized language of the system. The way in which the requirements were met in the creation of a language processor at the Institute of Cybernetics imeni V. M. Glushkov, Ukrainian Academy of Sciences, is discussed. The language processor utilizes a learning period during which a sequence of natural language statements must be input, each of which contains two significant words from a fixed set of semantic categories. A portion of the language database of the processor is a dictionary which is formed in the process of teaching the system by example. One distinguishing feature

is the process of automatic segmentation of word combinations into nouns and adjectives. It provides for automatic formation of lexical-grammatical information necessary for natural language processing. The semantic-syntactic analysis algorithm is briefly described. Semantic-syntactic analysis has been implemented in the form of a system of software written in PL-1 under OS YeS called the OLIMP [Russian acronym for learning linguistic multifunctional processor]. The system can be used as a mediator between a human user and an automated information processing system. If combined with the APROS scientific research automation system, the user can hold a dialogue with the planning system, formulate goals and receive results in a more convenient form. Figures 3; references 2 Russian.  
[196-6508]

UDC: 681.322.06:550.83

FIELD COMPUTER SYSTEM BASED ON 'ELEKTRONIKA 100-25' MINICOMPUTER AND 'ELEKTRONIKA MT-70' PERIPHERAL PROCESSOR

Novosibirsk AVTOMETRIYA in Russian No 6, Nov-Dec 84  
(manuscript received 21 Nov 83) pp 3-6

LARIN, F. I., MEZHOV, V. Ye., SAFIULLIN, I. G., SUBBOTIN, M. V. and TALOV, I. L., Voronezh-Novosibirsk

[Abstract] Seismic materials to be processed on a large computer are frequently preprocessed on smaller computers in the field to save mainframe machine time. By connecting a powerful minicomputer to peripheral devices for seismic information input and output, an independent processing system can be produced which can perform more than the usual preprocessing. The output of such a system might be considered either final output or intermediate data for deeper or specialized processing on the main computer. A configuration for a field computer system has been implemented by the author using an 'Elektronika 100-25' minicomputer, two 4.8 MB IZOT-1370 disk drives and two IZOT-5003 magnetic tape drives with enhanced controller to allow reading of gapless magnetic tape. An 'Elektronika MT-70' specialized high speed peripheral processor is used to accelerate processing and allow parallel processing of data obtained by other geophysical methods. Results can be output on magnetic tape and processing continued on large computers such as a BESM-6 or Cyber-73 at any point in the process. The Kvant-2 seismic information output device can handle 12-768 0.375-6 sec traces in one frame. References 6 Russian.  
[209-6508]

UDC: 681.142

AUTOMATED SYSTEM FOR MEASUREMENT, ANALYSIS AND PROCESSING OF EXPERIMENTAL DATA AT THE INSTITUTE OF HIGH ENERGY PHYSICS, KAZAKH ACADEMY OF SCIENCES

Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan 85  
(manuscript received 23 May 84; after revision 17 Aug 84) pp 90-94

TASHIMOV, M. A. and CHASNIKOV, I. Ya.

[Abstract] The Institute has utilized a measurement-computer system for about ten years. The system consists of eight semiautomatic measurement devices, an automatic scanner, spiral measurement device and semiautomatic television devices for measurement of ionization losses, as well as measurement projectors and semiautomatic microscopes. The computer system is a modernized BESM-6/7 plus two BESM-4 computers, a YeS-1010, 'Elektronika-100' and SM-4 small computers, modern IVK-1 and IVK-2 measurement computer systems plus about a dozen 'Elektronika-60' microcomputers. The measurement devices and computers are combined into several automated systems: the PA-BESM-4-I, SA-BESM-4-II, SPM-YeS1010 and other systems. Each of these systems is described, noting devices included and primary functions. The automated systems are used for measurement, analysis and processing of information produced in the process of experiments on the modern accelerators at the Institute. Figures 7; references 13 Russian.

[196-6508]

UDC: 519.95

ONE-DIMENSIONAL STATISTICAL ANALYSIS OF GEOMETRIC HEIGHTS ON GLIDE PATH

Tashkent PRIKLADNAYA MATEMATIKA I MEKHANIKA in Russian 1983  
(signed to press 20 Dec 83) pp 89-94

KHAIRAKULOV, I. V., SHEYMAN, A. V., YUGAY, L. P. and YANOVSKIY, Z. A.

[Abstract] The Computer Center of the Uzbek Civil Aviation Administration has developed an automated system for processing and analysis of flight information from flight recorders. Various statistical methods of data processing are applied to monitor the quality of pilot's work. This article presents an analysis of changes in the important parameter of geometric altitude to monitor aircraft control. Data from an IL-62M flight in 1981 are used to illustrate the analysis. References 2 Russian.

[176-6508]

UDC: 681.3.06: (528.8.04+53.082)

INFORMATION SUPPORT FOR SYSTEM FOR PROCESSING REMOTE AND CONTACT  
MEASUREMENTS OF ENVIRONMENTAL PARAMETERS

Baku IZVESTIYA AKADEMII NAUK AZERBAYDZHANSKOY SSR: SERIYA FIZIKOTEKHNICHESKIH  
I MATEMATICHESKIH NAUK in Russian No 3, 1984 (manuscript received 9 Nov 83)  
pp 103-107

ASKEROV, T. M. and AGAYEV, A. A., Scientific-Production Association for  
Space Research, AzSSR Academy of Sciences

[Abstract] A database including surface, remote and documentary data is described (BNDDD). The database is the information support subsystem for an automated system for processing of remote and ground surface data. It includes information, language and programming facilities designed to perform the functions of storage of information in the database, description of the structures of problem data and manipulation of these data. The basic languages used to manipulate the database are FORTRAN IV and PL/I. The basic languages are expanded to include data description and manipulation facilities, decreasing the dependence of application programs on data format. Application programs are included for the database administrator and for nonprogrammer-users. The database includes images of the surface of the earth obtained from flight vehicles, image descriptions, description of the placement of objects, phenomena and processes on the surface and object and phenomenon classifiers. Application programs are connected to the DBMS by inclusion of references to the resident DBMS module in application program source code. The module supports the description of maps or images, operations with files, creation of local databases and protection of data integrity. Experimental operation of the information support system at the computer center of the author's organization has shown its effectiveness and indicated areas for future improvement of the database support system.

References 3: 2 Russian, 1 Western.

[177-6508]

UDC: 519.12

THE 'SKALD' PROGRAM - EXPERIENCE IN MODELING POETIC CREATIVITY FOR A COMPUTER

Kiev KIBERNETIKA in Russian No 5, Aug-Sep 84  
(manuscript received '19/14/80' [sic]) pp 86-88

KONDRATOV, A. M., Leningrad Institute of Physical Culture imeni Lesgaft and  
ZUBOV, A. V., Minsk State Pedagogical Institute of Foreign Languages

[Abstract] The 'SKALD' program described in this article, developed in consultation with M. I. Steblin-Kamenskiy, differs from previous 'stochastic imitation' poetry-composing programs in that words are selected and texts

generated on the basis of rhythm, euphony and semantic parameters. Modeling the poetic creativity of the Scandinavian SKALD poets, who yielded poetry which has been justifiably recognized as the 'most difficult in the world' both in its pattern scheme, formal requirements and demands for phonic organization, the SKALD program utilizes a dictionary or data base to generate SKALD poetry at the computer center of Minsk State Pedagogical Institute of Foreign Languages. The data base consists of all one-syllable words in the Russian language, indexed by type of speech. The stages in generation of a poem include selection of the theme, selection of the genre, selection of the size, selection of rhythm-type and selection of individual randomization parameters. Sample Russian SKALD poems generated by the computer are presented. References 11 Russian.

[204-6508]

UDC 520.8

#### ANALOG AND DIGITAL ASTRONOMICAL IMAGE PROCESSING

Kiev VISNYK AKADEMIYI NAUK UKRAYINSKOYI RSR in Ukrainian No 2, Feb 85 pp 6-30

USYKOV, O. Ya., academician, Ukrainian SSR Academy of Sciences, DUDINOV, V. M., TSVYETKOVA, V. S. and SHKURATOV, Yu. H., candidates of physicomathematical sciences, and KORNIYENKO, Yu. V., PARUSYMOV, V. H. and STANKEVYCH, D. H.

[Abstract] A review is presented of the analog and digital technology currently employed by the observatory of Kharkov State University and other research establishments in the Ukraine in processing astronomical images. The computer-based methodology has provided new quantitative insight into the surface structure and origins of various celestial bodies, fine gravitational effects, and spectral classification of stars. The most often used technique consists of conversion of an image into a matrix of numbers for easy computer manipulation, analysis, and image recovery and enhancement. Considerable advancements have been made in speckle interferometry by employing fast Fourier transform algorithms. Much valuable data has also been obtained from iterative nonlinear techniques in which the computer provides best output image consistent with the quality of input. Specific information is included from Soviet probes on Mars, Venus and Jupiter. Figures 18; references 67: 6 Ukrainian, 51 Russian, 10 Western.

[256-12172]

NETWORKS

UDC: 681.324

INTEGRATED NETWORK SYSTEM FOR AUTOMATING SCIENTIFIC RESEARCH

Riga AVTOMATIKA I VYCHISLITEL'NAYA TEKHNIKA in Russian No 4, Jul-Aug 84  
(manuscript received 16 Jan 84; 27 Oct 83) pp 44-53

ZUDIN, O. S.

[Abstract] The scientific system on board the research vessel Akademik Boris Petrov is an integrated network automation system. The network system combines various types of measurement equipment, laboratory computers and the computer in the computer center, CAMAC apparatus, an automated weather station, an integrated navigation system, a multiple-beam and narrow-beam sonar system, a subsystem for collection of seismic data, deep-water soundings, various data transmission lines and interfaces with the local communications network. The software of the system includes the operating systems of the individual computers, application programs for scientific measurements and the networking software. The structure and basic organization principles of the network data collection and processing system on the vessel are discussed without presenting detailed discussion of special problems relating to integrating the system. The well-developed software, with standardized driver modules and simplified procedures for changing the functionality of modules by interchanging PROMs, allows easy expansion of the modular system. In the overall system diagram, three disk drives labeled "Winchester" are shown. The laboratory subsystems uses the VT20/X computer. Figures 5; references 7: 3 Russian, 4 Western.  
[137-6508]

UDC: 681.324

STRUCTURE OF HIGH-SPEED YeS COMPUTER-BASED WORKSTATION FOR A LOCAL AREA NETWORK

Riga AVTOMATIKA I VYCHISLITEL'NAYA TEKHNIKA in Russian No 4, Jul-Aug 84  
(manuscript received 5 Mar 84) pp 64-66

RED'KO, V. A. and GOBZEMIS, V. A.

[Abstract] Simulation modeling was used to compare a microprocessor-based workstation with a traditional common bus connecting the workstation to the

computer I/O channel and a microprocessor-based workstation with separate I/O and processing buses. The comparison showed that the structure with separate buses had a significant advantage in terms of throughput capacity. The evaluation was used as the basis for development of a microprocessor-based workstation with separate information transport and processing buses, direct memory access, parallel occurrence of information exchange with the computer, monochannel and processor, 16-bit memory organization and practically unlimited buffer memory capacity. The functions of the various modules of the workstation are briefly described. It was implied that the workstation was designed around one of the Elektronika series micros. Figures 2; references 1 Russian.

[137-6508]

UDC: 681.324

#### COMMUNICATIONS SUBNETWORKS OF LOCAL AREA COMPUTER NETWORKS

Riga AVTOMATIKA I VYCHISLITEL'NAYA TEKHNIKA in Russian No 6, Nov-Dec 84  
(manuscript received 19 Jun 84) pp 32-52

YAKUBAYTIS, E. A.

[Abstract] A communications subnetwork in a local area network includes all of the connectors and devices which provide the interconnection among the user machines on the network. This article provides general descriptions of the communications subnetwork, including types of channels and switching devices and their characteristics, and communications systems structures. Single-node switching systems, ring networks and networks of nodes arranged in a cellular pattern are included. Functions to be performed by the administrative and channel management systems are described. General advantages and disadvantages of types and arrangements of communications subnetworks are described. Figures 28; references 14: 2 Russian, 12 Western.

[142-6508]

UDC: 681.324

PRINCIPLES OF DESIGN OF LOCAL AREA NETWORKS USING UNIVERSAL MICROPROCESSORS

Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan 85  
(manuscript received 27 Feb 84) pp 35-39

MALINOVSKIY, B. N., NIKITIN, A. I., ZHAROVSKIY, S. N., ALISHOV, N. I. and SHALUGIN, S. S.

[Abstract] The development of microprocessor technology has permitted the creation of smart communications nodes which can adapt to a wide variety of data transmission systems. The design of such a communications node allows specialized or standardized adaptable low-level protocols to be created supporting full transparency of users and standardizing both logical and physical data transmission devices. This article describes the composition, structure and functional capabilities of a universal microprocessor-based communications node (MUS) designed to transmit data among terminals, computers and networks. The MUS is based on a type UVS-01 processor (K580IK80 plus 64 KBytes memory) of the 50-04 microprocessor debugging system, and allows organization of the first three levels of data interchange protocol (specialized and X.25). coordination of communications protocols among levels in various networks, organization of multimachine computer systems and the performance of smart terminal functions. The MUS includes networks and the UVS-91 and interface modules to coordinate transmission of data between computers, terminals and the MUS. Arrangements of MUS nodes in star, ring and sequential networks are described and diagramed. The principle suggested for design of LANs using the MUS modules allow the construction of networks differing in both configuration and purpose. Interface modules permit connection to YeS, BESM-6, SM, Elektronika, SO-04 and SOY-1 computers; to CO-01, UVV, and Elektronika-based terminals; and to various workstations (ARM). The first stage of a network of ARMs for scientific research has been created at the scientific-technical complex "Institute of Cybernetics imeni V. M. Gluskov" of the UkrSSR Academy of Sciences. Figures 3.

[196-6508]

THEORY OF COMPUTATIONS

UDC: 519.853

SEARCH FOR OPTIMAL GUARANTEED SOLUTIONS OF LARGE PROBLEMS IN SEMI-INFINITE PROGRAMMING

Moscow ZHURNAL VYCHISLITEL'NOY MATEMATIKI I MATEMATICHESKOY FIZIKI in Russian  
No 1, Jan 85 (manuscript received 2 Jul 81; after revision 18 May 84) pp 45-52

ZABOYEVA, O. A., TIMOKHIN, S. G. and SHAPKIN, A. V., Moscow

[Abstract] Problems in semi-infinite programming refer to problems of seeking the extremal of a linear function of a finite number of variables with an infinite number of linear constraints on the variables, i.e., problems in linear programming with an infinite number of constraints. A study is made of the most general type of problems of this class for which the set of matrices is a convex, closed, limited set. A previous work developed a theory of duality for such problems and linear methods of their solution. This work briefly presents the basic theoretical results and on their basis outlines a decomposition method of solving problems of linear programming with multiple conditions for the case of great dimensionality of the matrix of conditions. The search for an optimal guaranteed plan in a problem of semi-infinite programming by decomposition methods requires the creation of flexible software which can automate the preparation of initial data for the linear programming problems solved in the cycle. The authors know of no such software. Creation of this software will facilitate extension of the ideas for the search of guaranteed plans allowing the reliability of planning to be improved. Experimental testing of parts of the method was carried out on a YeS-1060 computer. References 8: 7 Russian, 1 Western.

[208-6508]

EDUCATION

COMPUTER USE IN GRADE SCHOOL CITED

Moscow SOVETSKAYA KULTURA in Russian 2 Apr 85 p 2

[Article by A. Andrusenko: "Computers In The School"]

[Text] As already reported in the press, the CPSU Central Committee Politburo has defined measures for comprehensively introducing computer equipment into the teaching process. Today we are recounting the experience of a Krasnoyarsk school where computers are being mastered successfully.

"Excuse me, you are illiterate!" We can actually hear this reproach even today. And reading quickly, writing beautifully or being able to quickly and beautifully put our ideas down on paper won't help us. By the way, what we are talking about is what is called computer literacy. But don't be in a hurry to brush aside the fact that they say we will survive without this and can manage as we did previously.

Today the computer calculates the optimal regimen for processing crankshafts, sells airline tickets and controls complicated equipment. The paradox is that the development of equipment has gone so far that it is simply unprofitable and illogical to use human brains to process the mass of work information.

Yet all the same, today the computer is still a "black box" for the majority of us. For many it is still easier to calculate their room account on an abacus than it is to learn how to work with a computer. And there is a reason for this. Those people who run into computers many years after they have left their educational institution have strong psychological biases against computers. But the scientific-technical revolution does not like to wait. The specialty with one of the worst shortages today is programming and the demand for it will continue grow like wildfire.

Children in the fourth grade of the 41st Middle School in Krasnoyarsk are not timid in front of a computer. The first task that the instructors give themselves is to train children to work freely with computers and to get used to using a computer in their studies as easily as they use a slide rule.

School children are being taught to program within the framework of an All-Union experiment to instill elements of school information structure in the

educational process. In the very near future the school expects delivery of the Agat micro-computer for the computer equipment class. And in the meantime they are using the full extent of the wonderful capabilities at the Krasnoyarsk branch of the USSR AN [Academy of Sciences] Siberian section's Computer Center.

They began training the kids to program in 1983, but even today the question for the school's teachers is not do they have to teach children to work with computers (they understand that it is necessary), but what should the method of teaching be. Teachers are interested in what the computer can bring to chemistry, physics, mathematics and even literature classes. They are taking the rich experience accumulated by an academician from Novosibirsk, Andrey Petrovich Yershov, who developed the teaching ideas and programs, as the basis of their work. One of the primary tasks is to arrange things so that the foundation of the schoolchildren's work is the logic of thought and not the technique of calculating.

Today introductions to this ultra-modern profession are taking place in the industrial training room. The fourth and fifth grades are busy with school information hobby groups. Beginning in the seventh grade, exercises are included in the academic plan. Only a small amount of time has passed since the beginning of the experiment, but the school is already coming up with some results. For example, dozens of the best students were in a competition for professional programmers that took place at the Computer Center (VTs) of Krasnoyarsk's Academgorodok. They not only coped with the assigned task: They even solved it in an original manner. VTs employees G. A. Shostak, G. P. Morozov and L. B. Chubarov, with the active participation of 41st School Director, Larisa Yanovna Kozlova, helped the children learn the capabilities of the equipment. But whereas VTs workers are interested in the school children primarily as future programmers and computer operators, teachers are interested in shorter-term problems.

Natal'ya Dmitriyevna Izmest'yeva teaches chemistry to senior class students and she feels that computer use could free up almost half of her lesson. The computer is radically changing the dynamics and content of mathematics lessons, for children can discover many mathematical laws themselves.

Students are no less excited by computer capabilities than are their teachers. Tenth grader Gleb Vorob'yev was among the participants in the programmer competition. He intends to link his future with the computer and today, while working at the VTs, he is assisting scientists to calculate the law of forest fire distribution by taking into account seasonal winds, humidity and a number of other factors.

Lena Golubtsova is busy calculating integrals and Sasha Orlov is studying the mathematical laws necessary to identify sets. All of this is a small part of the many real tasks that VTs employees are solving today. By helping them, children are learning the capabilities of the computer and the logic of thought and are also studying the basics of the programmer profession.

12511  
CSO: 1863/304

## CALL FOR MASS COMPUTER-LITERACY TRAINING IN LATVIA

Riga SOVETSKAYA LATVIYA in Russian 21 May 85 p 2

[Article by E. Yakubaytis, vice-president of the Latvian Academy of Sciences]

[Abstract] The author assesses the status of information-science technology in the Latvian republic, commenting on features of data banks which are in use and on advantages of modern data-processing and data-retrieval equipment.

The author calls for action to promote broader use of the latest technology. He points out that the great majority of the republic's computer centers are still using older-generation software, such as punch cards, perforated tape and lengthy printout sheets, and that they have no plans for introducing newer technology. The author proposes that computer-literacy courses and training in modern computer dialog techniques be introduced on a mass scale. He calls for compulsory courses of this type in programs for advanced training of specialists at the Latvian State University and the Riga Polytechnical Institute, and also in the republic Academy of Sciences. He recommends that the republic academy and State Planning Committee be put in charge of coordinating this mass training program, and that a television course, "Principles of Information Science and Computer Technology," be introduced, beginning with the next school year.

CSO: 1863/353  
FTD/SNAP

## ORGANIZATIONS

### UNESCO CENTER FOR ENGINEERING EDUCATION AUTOMATION IN TBILISI

Tbilisi ZARYA VOSTOKA in Russian 7 Jun 85 p 3

[Extract] An international center for modeling and automating the designing of management systems has been created in Tbilisi. It will operate at the chair of management information systems of the Georgian Polytechnical Institute. This center will become part of an international network for the employment of computers. It will coordinate work which is being done in this direction by major engineering schools of Bulgaria, Hungary, Great Britain, Denmark, Italy, France and the Federal Republic of Germany.

The institute concluded an agreement with UNESCO on carrying out a model experimental project on the topic "Microcomputers in Higher Technical Education," which became effective at the beginning of this year. The agreement calls for scientific research and planning work in the field of instruction, the use of national alphabets, development of methods for teaching engineers how to use equipment for computerized analysis and programming of management systems, etc. The purpose of the project is to prepare methodological materials and packages of applied programs for use in developing countries.

Professor Y. Menteshta, a prominent scientist and head of the information-science section of UNESCO's science sector, recently visited Tbilisi. In an interview, he told our correspondent: "I came for the purpose of discussing the center's program of work for the years immediately ahead, ascertaining the Georgian Polytechnical Institute's needs for equipment, and drafting a plan for the exchange of science associates. I was also interested in the status of your institute's developments in the field of information science.

"The model experimental project and the creation of the international center are the first projects to be carried out for UNESCO in the Soviet Union.

"The Georgian Polytechnical Institute is a major educational and research institution. Interesting work has been done here on the development of information systems for higher education and of decision-making systems, and also in the fields of artificial intelligence and machine designing of management systems, in particular.

"I should like to note that works by associates of this chair of instruction are well known abroad and are often published in scientific periodicals. This applies in particular to works by Professor G. Chogovadze, with whom we are now cooperating in UNESCO. He heads a section in our education sector.

"The holding of an international conference on questions of automated designing of management systems is planned in Tbilisi in 1987."

CSO: 1863/353

FTD/SNAP

- END -

**END OF**

**FICHE**

**DATE FILMED**

September 17, 1985